

# Redaxo-Dokumentation

## Versionen 3.0 - 3.2



Verfasser: Peter Wolfrum  
Dagmar Uttich  
Markus Staab

Stand: 26. Juni 2006

# Inhaltsverzeichnis

|  |    |
|--|----|
| Dokumentation REDAXO 3.0, 3.1, 3.2.....      | 5  |
| A. Erste Schritte.....                       | 5  |
| A.0 Über Redaxo.....                         | 5  |
| A1. Installation.....                        | 7  |
| A.1.01 Download.....                         | 7  |
| A.1.02 Systemvoraussetzungen .....           | 7  |
| A.1.03 Upload .....                          | 8  |
| A.1.04 Setup .....                           | 8  |
| A1.05 Mehrfachinstallation von Redaxo.....   | 13 |
| A1.06 Sicherheitshinweise.....               | 13 |
| A.2 Demoseiten .....                         | 14 |
| A.3 Webseiten erstellen mit Redaxo.....      | 16 |
| B. Redaktion .....                           | 19 |
| B1. Strukturverwaltung .....                 | 20 |
| B1.01 Kategorien .....                       | 21 |
| B1.02 Artikel .....                          | 22 |
| B1.03 Block/Modul .....                      | 24 |
| B2. Medienpool .....                         | 26 |
| B3. Statistiken .....                        | 29 |
| C. Entwickler & Admin .....                  | 30 |
| C1. Templates .....                          | 31 |
| C1.01 Template erstellen .....               | 31 |
| C1.02 Navigation .....                       | 32 |
| Download von Navigationen.....               | 33 |
| C1.02.01 Navigation .....                    | 34 |
| C1.03 Mehrere Artikel einbinden .....        | 35 |
| C1.04 Seitenspezifische Metadaten .....      | 36 |
| C1.05 Includes .....                         | 39 |
| C1.06 CTYPES .....                           | 40 |
| C2. Module & Aktionen .....                  | 42 |
| C2.01 Modul erstellen und bereitstellen..... | 43 |
| C2.02 REDAXO Variablen.....                  | 44 |
| C2.03 Actions.....                           | 46 |
| C3. Benutzer .....                           | 49 |
| C3.01 Rechtevergabe Version 3.0, 3.1.....    | 50 |
| C3.02 Rechtevergabe Version 3.2.....         | 52 |

|   |    |
|---|----|
| C4. Addons .....                            | 55 |
| C4.01 Struktur.....                         | 55 |
| C5. Specials .....                          | 57 |
| C5.01 Main Preferences (Version 3.0).....   | 57 |
| C5.02 Sprachen (ab Version 3.0).....        | 59 |
| Inhalte von Iso auf Utf-8 konvertieren..... | 60 |
| C5.03 Typen (ab Version 3.0).....           | 61 |
| C6. Object Oriented Framework .....         | 62 |
| C6.01 OOCategory.....                       | 63 |
| OOCategory-Klasse.....                      | 63 |
| getCategoryById(2).....                     | 63 |
| searchCategoriesByName().....               | 64 |
| getRootCategories().....                    | 65 |
| getChildren().....                          | 67 |
| getArticles().....                          | 69 |
| getStartArticle().....                      | 70 |
| getId().....                                | 71 |
| getName().....                              | 72 |
| getPriority ().....                         | 72 |
| getUrl().....                               | 73 |
| isOnline ().....                            | 73 |
| toString().....                             | 74 |
| C6.02 OOArticle.....                        | 75 |
| OOArticle-Klasse.....                       | 75 |
| getArticleById().....                       | 75 |
| searchArticlesByName().....                 | 77 |
| getArticlesByType(id).....                  | 78 |
| getCategory().....                          | 80 |
| getSlicesOfType().....                      | 80 |
| getId().....                                | 81 |
| getName().....                              | 81 |
| getDescription().....                       | 82 |
| getUrl().....                               | 83 |
| isOnline().....                             | 84 |
| toString().....                             | 84 |
| C6.03 OOArticleSclice.....                  | 85 |
| OOArticleSlice-Klasse.....                  | 85 |
| getArticleSliceById().....                  | 86 |
| getFirstSliceForArticle().....              | 87 |
| getSlicesForArticleOfType().....            | 87 |
| getNextSlice().....                         | 88 |
| getPreviousSlice().....                     | 90 |
| getArticle().....                           | 91 |
| getId().....                                | 92 |
| getValue.....                               | 92 |

|                               |     |
|-------------------------------|-----|
| getLink().....                | 93  |
| getLinkUrl().....             | 93  |
| getFile().....                | 94  |
| getFileUrl().....             | 94  |
| getHtml().....                | 95  |
| getPhp().....                 | 95  |
| C6.04 OOMedia.....            | 96  |
| OOMedia-Klasse.....           | 97  |
| getMediaById().....           | 97  |
| getMediaByName().....         | 99  |
| getMediaByFileName().....     | 100 |
| searchMediaByFileName().....  | 101 |
| getMediaByExtension().....    | 102 |
| searchMediaByExtension()..... | 103 |
| C6.05 OOMediaCategory.....    | 104 |
| OOMediaCategory-Klasse.....   | 105 |
| C7. Import & Export .....     | 110 |
| C8. CVS.....                  | 111 |
| C.9 Extension Points.....     | 113 |
| C9.02 Beispielcode.....       | 113 |

# Dokumentation REDAXO 3.0, 3.1, 3.2

## A. Erste Schritte

Auf den folgenden Seiten dieses Kapitels erfahren Sie mehr über die Installation von Redaxo auf dem Webserver Ihrer Wahl.

Überprüfen Sie vor Beginn die [Systemvoraussetzungen](#) für Ihr Webpaket und lernen Sie mehr über die ersten Schritte mit einem flexiblen Content-Management-System.

### Links zu Redaxo:

**Redaxo.de** | <http://www.redaxo.de>

Aktuelle Informationen zu den aktuellen Versionen, die Basis-Installation und Updates erhalten Sie auf der offiziellen Redaxo-Webseite.

**Forum.Redaxo.de** | <http://forum.redaxo.de>

Eine der ersten Anlaufstellen für Support, Fragen, Tipps und "Insider-Wissen" erhalten Sie im Forum zu Redaxo.

## A.0 Über Redaxo

Unser Ziel war und ist ein einfaches, schnell zu erlernendes Redaktionssystem zu entwickeln, welches dennoch einen hohen Grad an Flexibilität gewährleistet. Wir hoffen, eure Zustimmung zu finden, dass uns dies geglückt ist. Wir nehmen auch gerne Anregungen von euch entgegen, um Redaxo kontinuierlich zu optimieren.

Redaxo bietet die Möglichkeit, Webseiten zu erstellen, die hinsichtlich der individuellen Gestaltung keinerlei Einschränkungen unterliegen. Gleichzeitig wird durch die Trennung von Inhalt, Funktionalität und Design eine leichte und schnelle Bearbeitung gerade auch für mehrere Bearbeiter mit unterschiedlichen Kenntnissen gewährleistet. Die Aktualisierung von Inhalten kann anschließend ohne besondere Programmierkenntnisse erfolgen.

Ein Vorteil von Redaxo ist, dass bereits HTML/CSS-Kenntnisse ausreichen, um Internetauftritte mit Redaxo realisieren zu können. Alle Anpassungen an spezielle Anforderungen können in PHP, Javascript usw. programmiert werden. Eine zusätzliche Skriptsprache ist bei Redaxo nicht vorgesehen.

Die bei Redaxo mitgelieferten Demo-Versionen erleichtern den Einstieg und bieten einen ersten Eindruck über die Webseitenerstellung mit Redaxo.

## Standard-Funktionen

Das Redaxo-Backend ist einfach aufgebaut und klar strukturiert. Es bietet eine Reihe von Standard-Funktionalitäten:

- x Die **Strukturverwaltung** ermöglicht die Verwaltung der Kategorien und Inhalte
- x Über den **Medienpool** werden relevante Dateien - Grafiken, Skripte, Videos u. ä. - hochgeladen und verwaltet.
- x Es ist eine **Benutzerverwaltung** integriert, über die einzelnen Personen ein zielgerichteter und ggf. auch eingeschränkter Zugriff auf das Backend ermöglicht wird.
- x Es gibt jeweils einen Bereich für **Templates** und **Module**.
- x Die **Statistikfunktion** gibt Auskunft zum Besucherverhalten
- x Die **Import/Export-Funktion** ermöglicht Datensicherung und Aktualisierung von Daten.
- x Der **Linkchecker** prüft auf korrekte Verlinkung.
- x Im Bereich Sprache läßt sich **Mehrsprachigkeit** leicht installieren.
- x Der **WYSIWIG-Editor** TinyMCE kann als Modul eingebunden werden.

## Erweiterungen

Alle weiteren Funktionen können als AddOns oder Module in das System integriert werden. Im Downloadbereich von Redaxo gibt es eine ausführliche Sammlung mit der entsprechenden Dokumentation.

Beispiele für Erweiterungen:

- x Newsletter
- x diverse Bildergalerien
- x Formularfunktionen
- x Gästebuch
- x Bread Crumb Navigation
- x Sitemap-Erstellung
- x Google Sitemap Generator
- x url-rewrite-Funktionen
- x Adressen-Modul
- x Email-Adressen-Schutz
- x RSS-Reader
- x Suchfunktion

Es gibt noch viele weitere AddOns und Module und dank der sehr aktiven Community werden ständig neue ergänzt. Es lohnt sich, immer mal wieder einen Blick in den Downloadbereich zu werfen.

# A1. Installation

In den folgenden Kapiteln werden die ersten Schritte sowie die Installation von REDAXO beschrieben. Nach dem erfolgreichen Setup können Sie anhand der mitgelieferten Demos und diesem Manual das Konzept, Abläufe und Funktionen von REDAXO kennenlernen.

## A.1.01 Download

Auf der offiziellen Redaxo-Webseite erhalten Sie die [aktuelle Basisversion](#) sowie Updates oder ältere Versionen:

Laden Sie die gewünschte Version herunter und entpacken Sie das Zip-Archiv. Die entpackte Ordnerstruktur sollte nun in der folgende Struktur vorliegen:

### REDAXO 3.x

```
/files  
/js  
/pics  
/redaxo  
_htaccess  
index.php
```

Die komplette Ordnerstruktur sollte für die Installation nicht verändert werden. Einen Überblick über die komplette Ordnerstruktur von Redaxo, wird im Kapitel [Installation – Upload](#) beschrieben.

### Kurze Beschreibung:

Im Ordner **files** werden alle Dateien gespeichert, die in Redaxo über den Medienpool auf den Server geladen werden. Die Ordner **css**, **js** und **pics** dienen zur Strukturierung Ihrer Webseite.

Die Datei **index.php** dient zur Darstellung aller Artikel/Seiten der Webseite.

## A.1.02 Systemvoraussetzungen

Folgende Voraussetzungen werden von REDAXO an ihren Webservice gestellt:

- PHP >= 4.2.0, [5.1]
- MySQL 3, 4, [5]
- FTP Zugangsdaten (Host, Login, Passwort)
- MySQL Zugangsdaten (Host, Datenbank, Login, Passwort)

## A.1.03 Upload

Laden Sie nun den entpackten Ordnerinhalt der Basis-Version auf Ihren Webservice. Im allgemeinen empfehlen wir dafür das Root-Verzeichnis Ihres Webspaces. Für die notwendigen Einstellungen zum Upload auf Ihren Webserver, schlagen Sie in der Hilfe Ihres FTP-Programms nach.

Anpassungen um Redaxo in einem Unterverzeichnis zu betreiben, erfahren Sie Bereich D. FAQ.

Wenn Sie die komplette Struktur auf Webservice geladen haben, können Sie nun die Installation starten.

Einer der ersten Überprüfungen gilt den Rechten mehrerer Ordner und Dateien. Sie können die Installation starten und auf die mögliche Fehlerliste warten, oder gleich die notwendigen Einstellungen über Ihr FTP-Programm überprüfen und anpassen.

Folgende Ordner und Dateien benötigen die genannten Rechte:

### Version 3.x

| Rechte | Ordner   |
|--------|--|
| 775    | <b>/files</b>                                      |
| 775    | /redaxo/include/addons/import_export/ <b>files</b> |
| 775    | /redaxo/include/ <b>generated</b>                  |
| 775    | /redaxo/include/generated/ <b>articles</b>         |
| 775    | /redaxo/include/generated/ <b>cache</b>            |
| 775    | /redaxo/include/generated/cache/cache.php          |
| 775    | /redaxo/include/generated/ <b>files</b>            |
| 775    | /redaxo/include/generated/ <b>categories</b>       |
| 775    | /redaxo/include/generated/ <b>templates</b>        |
| 775    | /redaxo/include/ <b>install</b>                    |
| 775    | /redaxo/include/clang.inc.php                      |
| 775    | /redaxo/include/ctype.inc.php                      |
| 775    | /redaxo/include/addons.inc.php                     |
| 775    | /redaxo/include/functions.inc.php                  |
| 775    | /redaxo/include/master.inc.php                     |

**Die hier aufgeführten Berechtigungen (775) können je nach Serverkonfiguration unterschiedlich sein.**

## A.1.04 Setup

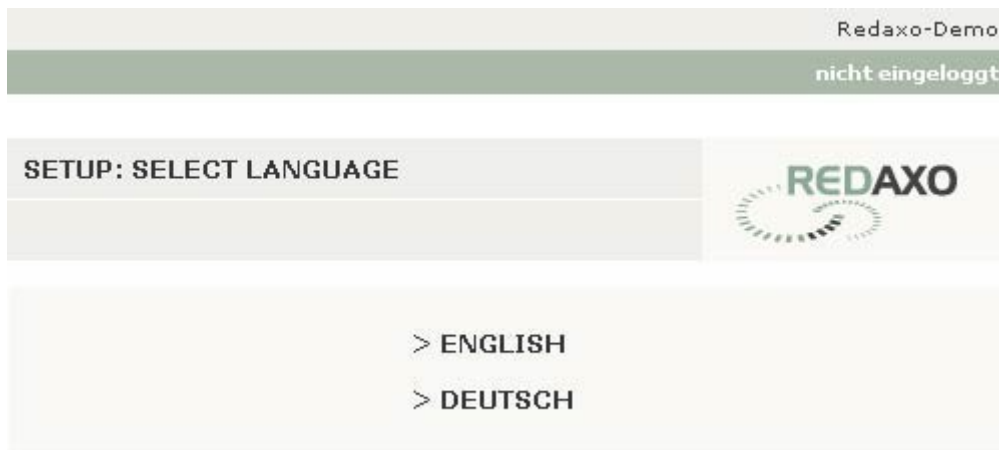
### Start

Um die Installation von Redaxo zu starten, geben Sie die URL Ihres Webspace ein und fügen an den Pfad

/redaxo oder

/redaxo/index.php an.

Redaxo benötigt die Einstellung **Magicquotes on**.



### Sprachversion

Wählen Sie danach die gewünschte Sprachversion aus und bestätigen Sie, dass Sie die Lizenzbestimmungen gelesen haben und starten danach das eigentliche Setup.

### Setup - Schritt 1

Auf der folgenden Seite wird die PHP-Version und Rechteüberprüfung alle Ordner und Dateien überprüft. Passen Sie mit Ihrem FTP-Programm ggfs die Rechte wie angegeben an.



### Mögliche Fehlermeldungen:

Fehler:

Die Datei / Das Verzeichniss ../redaxo/include/addons/import\_export/files existiert nicht, bitte erstellen Sie es.

Lösung:

Der Ordner kann bei der Installation/Upload nicht auf Server kopiert worden sein. Erstellen Sie im Ordner "redaxo/include/addons/import\_export/" einen Ordner "files" (Rechte: 777)

Und klicken Sie auf den Link ""Weiter mit Schritt 2""

Erscheint bei beiden Überprüfungen **ok** gehen Sie weiter zu Schritt 2.

## Setup - Schritt 2

Im nächsten Schritt werden wichtige Einstellungen in die **'include/master.inc.php'** geschrieben. Halten Sie hierfür die Angaben Ihres Providers bereit.

Redaxo-Demo  
nicht eingeloggt

SETUP: SCHRITT 2 von 5



**Schreiben der 'include/master.inc.php'**

```
// ---- Allgemein Redaxo Einstellungen
```

|                                 |   |
|---------------------------------|---|
| Serverdomain [optional]         | <input type="text" value="redaxo.com"/>               |
| Serverbezeichnung [optional]    | <input type="text" value="Redaxo-Demo"/>              |
| Fehler E-Mailadresse [optional] | <input type="text" value="jan.kristinus@pergopa.de"/> |

```
// ---- Datenbankinformationen
```

|                    |                          |
|--------------------|--------------------------|
| Name der Datenbank | <input type="text"/>     |
| MySQL Host         | <input type="text"/>     |
| Login              | <input type="text"/>     |
| Passwort           | <input type="password"/> |

## Setup - Schritt 3

Mit diesem Schritt werden die Tabellen in der Datenbank neu angelegt und je nach Auswahl der Option **überschrieben!**

Redaxo-Demo  
nicht eingeloggt

SETUP: SCHRITT 3 von 5

**REDAXO**

**Datenbank anlegen**

- Datenbank einrichten
- Datenbank einrichten und alte überschreiben falls vorhanden  
[**Vorsicht** - Alte Seite wird komplett gelöscht]
- Datenbank existiert schon [Weiter ohne Datenbankimport]

Weiter zu Schritt 4

Für den ersten Start von Redaxo wählen Sie die Option **Datenbank einrichten**. Damit werden alle relevanten Tabellen die Redaxo benötigt erstellt.

Sollten Sie phpMyAdmin zur Verfügung haben, können Sie die erstellten Tabellen von Redaxo überprüfen.

Wollen Sie Ihre Datenbank nicht überschreiben wollen, dann wählen Sie Option **Datenbank existiert schon [Weiter ohne Datenbankimport]**.

Folgendes Tabellen sollten nach der Basis-Installation in den Datenbank erstellt worden sein:

```
rex_action
rex_article
rex_article_slice
rex_article_type
rex_clang
rex_file
rex_file_category
rex_help
rex_module_action
rex_modultyp
rex_template
rex_user
```

#### Setup - Schritt 4

Hier legen Sie den ersten Benutzer für Ihre Redaxo-Version fest. Mit diesen Login-Daten können Sie sich in die Redaxo -Version einloggen. Sie können als Administrator zu einem späteren Zeitpunkt die Angaben zu diesem Account ändern, löschen und neu anlegen.

Redaxo-Demo  
nicht eingeloggt

SETUP: SCHRITT 4 von 5



**Administrator anlegen**

Login:

Passwort:

Keinen User anlegen

### Setup - Schritt 5

Wenn Sie auf dieser Seite angekommen sind, haben Sie es geschafft. Redaxo ist installiert und bereit für den ersten Login.

Folgen Sie dem Link oder geben Sie wie bei jedem Login die **Url/redaxo** in ihren Browser ein.

Redaxo-Demo  
nicht eingeloggt

SETUP: SCHRITT 5 von 5



Herzlichen Glückwunsch zu Ihrem REDAXO!

Bitte noch dieses beachten:

1. Mit dem eingerichteten Zugang einloggen **login**
2. Importieren einer Demo über [wenn eingeloggt]: Import/Export und dort jeweils die Datenbank und Files installieren.

Viel Spass und Erfolg

Das REDAXO Team

## A1.05 Mehrfachinstallation von Redaxo

Einem häufig geäußertem Wunsch der Community entsprechend, ist es seit Version 3.2 möglich, Redaxo mehrfach in einer Datenbank zu installieren. Die Unterscheidung der Installationen erfolgt durch Setzen unterschiedlicher Tabellen-Präfixe.

In der Datei `redaxo/include/master.inc.php` ist der Tabellen-Präfix standardmäßig auf `"rex_"` gesetzt.

```
$REX['TABLE_PREFIX'] = "rex_";
```

Dieser Wert muß vor der Installation einer weiteren Redaxo-Version manuell geändert werden. Der Wert wird beim Setup nicht abgefragt. Wird der Wert nicht vorab geändert, werden die Tabellen der vorhergehenden Installation überschrieben.

Noch sind nicht alle Addons auf diese Einstellung hin angepasst, deshalb ist dieses Feature mit Vorsicht zu genießen!

Weiterhin ist zu beachten, dass erstellte Exporte (Import/Export-Addon) nur in Installationen importiert werden können, die mit dem gleichen `TABLE_PREFIX` konfiguriert sind, wie diejenige, unter der der Export erstellt wurde!

## A1.06 Sicherheitshinweise

Aus aktuellem Anlaß möchten wir darauf hinweisen das der `"redaxo/include"`-Ordner vor Zugriff von aussen geschützt werden muss. Wir haben standardmäßig eine `".htaccess"`-Datei eingebaut, so dass Apache-Server im Normalfall keinen Zugriff auf die Dateien im Include-Ordner erlauben. Bei anderen Servern funktioniert dieser "Schutz" nicht und man muss dies entsprechend der Servermöglichkeiten selbst einstellen..

Allgemein: Der Zugriff auf den include Ordner muss so gesperrt werden das man über die Webseite nicht darauf zugreifen kann!

### **Eine Bitte an unsere Community**

Sofern Sicherheitslücken vorhanden sein sollten würden wir uns einen direkten Kontakt vorher wünschen, bevor diese Informationen ins Forum gestellt werden.

## A.2 Demoseiten

Ein erste Weg sich mir Redaxo vertraut zu machen, ist die Möglichkeit die mitgelieferten Demos zu installieren. In den Templates und Modulen finden Sie einen vollständige Webseite, die Ihnen die Arbeitsweise und die Möglichkeiten von Redaxo aufzeigt.

Um die mitgelieferten Demos zu installieren, wechseln Sie über die Navigation in den Bereich Import/Export.

Ein Import besteht in der Regel aus dem Import der Datenbank (Tabellen und Inhalte) und den Dateien, die beim Export erstellt wurden. Dabei werden in der Regel die Dateien aus den Ordnern files, css, js und pics gesichert. (Mehr unter Export)

### Import

Auf den linken Seite können Sie unter Datenbankimport die Tabellen und deren Inhalte importieren. Dazu können Sie entweder die Redaxo-Import Dateien auswählen und importieren, oder die darunter angegebenen Importe importieren. Dabei werden die bisherigen Inhalte der Datenbank gelöscht und überschrieben!

Danach können Sie unter Dateienimport die benötigten Dateien importieren. Nach erfolgreichem Import werden alle Artikel sowie der Cache neu generiert. Wechseln Sie danach in die Strukturverwaltung oder auf die Webseite und sehen das Ergebnis.

### Export

Auf der rechten Seite finden Sie Einstellungen und Optionen für den Export der Tabellen und Dateien ihrer Webseite.

Für Sicherung der Datenbank, wählen Sie erste Option und entscheiden unten, ob Sie die Export-Datei auf dem Server speichern oder auf Ihrer Festplatte sichern wollen.

Wählen Sie die Option Auf dem Server speichern wird die Datei auf dem Server gespeichert und auf der linken Seite, unter \*Import \*angezeigt. Sie finden die auf dem Server gespeichert Datei im Ordner "redaxo/include/addons/import\_exportfiles".

### Demoseiten von Redaxo 3.0

Bei der Version Redaxo 3.0 werden drei verschiedene Demoseiten mitgeliefert.

**r3\_demo1** ist ein kleine Demo die verdeutlichen soll, wie Redaxo im allgemeinen arbeitet.

**r3\_demo2\_multilang** ist die gleiche wie die erste, mit dem Unterschied, dass hier die Mehrsprachigkeit eingearbeitet wurde.

**r3\_demoshop** ist die Shop Variante. Dazu muss man das Addon Simple\_shop installiert und aktiviert haben. Diese Variante basiert auch auf Redaxo nutzt aber das Addon Simple\_Shop.



**Import**

Bitte bedenken Sie das beim Importieren einer Datenbank **die alte Datenbank gelöscht** wird. Bitte speichern/exportieren Sie diese vorher wenn Sie sich unsicher sind.

Datenbankimport

| Dateiname                  | Erstellt am         |        |         |
|----------------------------|---------------------|--------|---------|
| r3_demoshop_001.sql        | 07.08.2005<br>16:54 | Import | Löschen |
| r3_demo2_multilang_001.sql | 07.08.2005<br>16:40 | Import | Löschen |
| r3_demo1_001.sql           | 07.08.2005<br>16:38 | Import | Löschen |

Abb.: V3-demoversionen.gif

## A.3 Webseiten erstellen mit Redaxo

Redaxo basiert auf PHP und Mysql. Kenntnisse in dieser Sprache und im Umgang mit der Datenbank sind zwar zu empfehlen, aber nicht unbedingt erforderlich. Anhand der Demo-Versionen kann man bereits eigene Webseiten erstellen und dabei lernen, das System zu nutzen.

Weitergehende Anpassungen an spezielle Anforderungen können mit PHP/HTML-Kenntnissen problemlos vorgenommen werden.

Viele Fragen und Probleme können mit Hilfe des Forums, des Wikis oder dieser Dokumentation bearbeitet werden. Man kann hier auf einen umfangreichen Fundus an Problemlösungen zurückgreifen.

Wenn man Redaxo ohne Demo-Versionen auf seinem Server installiert hat und nun eine Webpräsentation erstellen möchte, sind die folgenden Schritte durchzuführen. Diese Beschreibung ist nur eine Kurzfassung. Die Details sind in den Folgekapiteln beschrieben.

### Schritte zur Erstellung einer Webseite mit Redaxo

1. Definition der Kategorien und Artikel
2. Erstellen eines Templates
3. Erstellen eines Stylesheets
4. Erstellen der seitenspezifischen Inhalte der Artikel (Blöcke und Module)
5. Festlegung eines Startartikels

### 0. Redaxo-Begriffe

Bei der Arbeit mit Redaxo werden einem die folgenden Begriffe immer wieder begegnen::

Kategorie

Artikel

Template

Module

Block/Slice

### 1. Definition der Kategorien und Artikel

Die einzelnen Seiten einer Präsentation werden bei Redaxo „Artikel“ genannt. Diese müssen unter der Rubrik „Struktur“ erzeugt und benannt werden. Zur besseren Gliederung und um die Navigation zu steuern, können die Artikel unterschiedlichen Kategorien zugewiesen werden. Dazu definiert man zunächst die Kategorie und legt dann in den Kategorien die zugehörigen Artikel an.



Abb.: strukturverwaltung.gif

### Erstellen eines Templates

Damit ein Artikel angezeigt werden kann, braucht er ein Template, das ihm zugewiesen wird. Ein Template ist eine html-Datei ohne seitenspezifischen Inhalt. Es legt das Layout des Artikels fest. Templates werden in der eigenen Rubrik „Templates“ definiert und gespeichert. Die Zuweisung zu einem Artikel erfolgt in der Rubrik „Struktur“.



Abb.: templates.gif

### Erstellen eines Stylesheets

Üblicherweise wird die komplette Formatierung einer Webseite in einem Stylesheet ausgelagert. Das wird auch für die Arbeit mit Redaxo empfohlen. Das Stylesheet wird wie sonst auch in html-Seiten in den head-Bereich des Templates eingebunden.

### Erstellen der seitenspezifischen Inhalte für einen Artikel (Blöcke und Module)

Die Seiteninhalte werden in einzelnen Blöcken (auch Slices genannt) eingegeben. Die Anzahl der Blöcke ist beliebig. Bei dem Anlegen eines Blocks muß man ein Modul wählen, mit dem der Block editiert werden soll. Module sind „Mintemplates“, die den jeweiligen Inhaltsbereich formatieren. Die Eingabe von Inhalten mittels Modulen erleichtert die Einhaltung eines durchgängigen Layouts.

Bei der „Blanko-Version“ von Redaxo sind noch keine Module enthalten. Sie können aber aus den Demoversionen oder von der Redaxo-Seite übernommen werden.

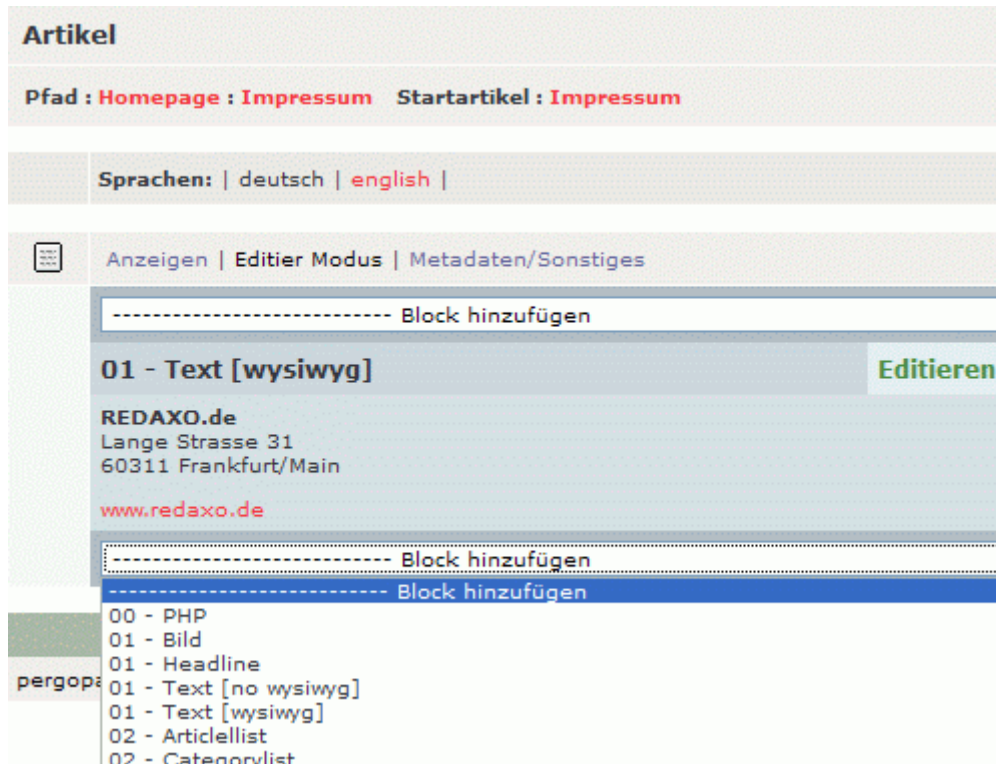


Abb.: artikel-editieren.gif

### Festlegung eines Startartikels

Wenn die genannten Schritte erledigt sind, d. h., die Struktur ist festgelegt, Templates und Stylesheet sind erstellt und zugewiesen und die Blöcke der Artikel sind mit Inhalten gefüllt, dann braucht man nur noch einen Startartikel zu benennen. Dies geschieht unter der Rubrik „Specials“. Standardmäßig ist der Artikel mit der Artikel-ID 1 als Startartikel vorgesehen. Hier wird die ID des gewünschten Startartikels eingetragen. Wenn man nun die URL aufruft, sollte dieser Artikel angezeigt werden.



Abb.: startartikel.gif

## B. Redaktion

Mit der Redaktion einer Webseite konzentrieren Sie sich auf die inhaltliche Verwaltung der Webseite.

Der Redakteur erhält in den meisten Fällen die Programmteile in REDAXO, die er für die Aufgabe benötigt. Alle anderen Teile werden dabei ausgeblendet. Die Einschränkung der Benutzerrechte können individuell vom Administrator oder Entwickler der Webseite vergeben werden.

In den folgenden Abschnitten finden Sie Informationen zur Redaktion einer Webseite in REDAXO.

### **B1. Strukturverwaltung**

Nach dem erfolgreichen Login in REDAXO ist die Strukturverwaltung die erste Seite die Sie sehen. Die Strukturverwaltung ist der zentrale Bereich.

### **B2. Medienpool**

Über den Medienpool können Dateien auf den Server geladen, verwaltet und gelöscht werden. Im allgemeinen werden darüber Grafiken und Textdateien (z. B. Stylesheets) verwaltet.

### **B3. Statistiken?**

Redaxo bietet umfangreiche Statistikfunktionen, mit denen man das Besucherverhalten für einen Webauftritt analysieren kann.

### **B4. Community?**

REDAXO bietet eine integrierte Verwaltung einer Community auf der verwalteten Webseite an.

Die in REDAXO weiterführenden, integrierte Programmteile wie

[Templates](#)

[Module](#)

[Import & Export](#)

[Addons](#)

werden im Bereich [C. Entwickler & Admin](#) beschrieben

# B1. Strukturverwaltung

## Artikel und Kategorie

Nach dem erfolgreichen Login in REDAXO ist die Strukturverwaltung die erste Seite die Sie sehen. Die Strukturverwaltung ist der zentrale Bereich.

Das seitenbildende Element ist der **Artikel**. Er steht in der Regel für eine Inhaltsseite ihrer Webseite. Für jeden Artikel kann ein **Template** ausgewählt werden, der den Rahmen oder das Erscheinungsbild der Webseite bestimmt. Über die Funktion **Status** kann soweit programmiert, die Seite **online und offline** geschaltet werden.

| Strukturverwaltung                   |                       |      |                     |                     |                 |         |        |
|--------------------------------------|-----------------------|------|---------------------|---------------------|-----------------|---------|--------|
| Pfad : <b>Homepage</b> : <b>HOME</b> |                       |      |                     |                     |                 |         |        |
|                                      | Kategorie             | Prio | Kategorie editieren |                     | Status/Funktion |         |        |
| ..                                   |                       |      |                     |                     |                 |         |        |
|                                      | Artikelname           | Prio | Template            |                     | Status/Funktion |         |        |
|                                      | <b>Home</b>           | 1    | 01 Standard         | <b>Startartikel</b> | ändern          | Löschen | online |
|                                      | <b>Nutzungsbeding</b> | 2    | 01 Standard         | <b>Artikel</b>      | ändern          | Löschen | online |
|                                      | <b>Registrierung</b>  | 3    | 01 Standard         | <b>Artikel</b>      | ändern          | Löschen | online |

Abb.: artikel.gif

Die Verwaltung der Artikel erfolgt über die **Kategorien**. In REDAXO wird eine hierarchische Struktur eingesetzt. D.h. Verschiedene Kategorien (Ordner) enthalten verschiedene Artikel (Inhalte). Jede Kategorie hat einen **Startartikel** (Startartikel sind Einstiegsseiten einer Kategorie- im Normalfall Übersichtsseiten ).

Normale Artikel können in beliebiger Anzahl erstellt werden. Einem Artikel kann man ein Template zuweisen welches die Darstellungsform bestimmt. Ein Artikel besteht aus mehreren **Blöcken** und repräsentiert den eigentlichen Inhalt. Die Erstellung dieser Blöcke basiert auf **Modulen**. Mittels der Module werden Eingabemasken für Textbausteine, Bilder u. a. definiert und die Anzeige der dort eingegebenen Inhalte formatiert.

Weiterhin hat der Artikel Metadaten die ihn allgemein beschreiben (z.B. Kurzbeschreibung, Suchbegriff und Grafik).

Kategorien können auch in der Form genutzt werden, um Navigationsstrukturen abzubilden. Für die meisten Fälle gilt: Was in der Struktur zu sehen ist, sieht man auch auf der Sitemap und in der Navigation. Die Priorität organisiert die Reihenfolge.

## B1.01 Kategorien

Kategorien werden, wie in einem Explorer oder Finder, zur Strukturierung und Verwaltung der Artikel erstellt.

### Erstellen (1)

Zum Erstellen einer neuen Kategorie, klicken Sie auf (+)-Symbol und geben Sie den Namen der Kategorie ein und speichern Sie Ihre Eingabe über die Schaltfläche "edit\_category" ab. Die so ersellte Kategorie ist in den **Funktion offline** gestellt. Für die Verwaltung hat die Einstellung keinerlei Funktion.

### Funktion/Status (2)

Diese Einstellung kann vom Entwickler ausgelesen und genutzt werden. Neben der Verwaltung können Kategorien die Seitennavigation Ihrer Webseite bilden. Über die oben beschriebene Funktion "online/offline" kann der Name sichtbar geschaltet werden. Diese Funktion wird in der Regel bei der Programmierung eines Templates genutzt.

Weitere Information hierzu finden Sie im Bereich [C. Entwickler & Admin/C1.02 Navigation](#)

Jede Kategorie enthält nach dem Erstellen immer einen Startartikel, der nicht gelöscht oder in der **Funktion offline/online** verändert werden kann.

Wechseln Sie in die gewünschte Kategorie, indem Sie auf den Namen der Kategorie klicken. Die Kategorie wird danach mit den enthaltenen Artikeln angezeigt.

### Prio (3)

Über die Einstellung Prio kann die Reihenfolge der Kategorien in der Strukturverwaltung verändert werden. Die Reihenfolge kann auch bei der Ausgabe der Kategorien auf der Webseite genutzt werden.

| Strukturverwaltung  |      | REDAXO                      |                   |  |
|---|------|-----------------------------|-------------------|--|
| Pfad : <b>Homepage</b>  |      |                             |                   |  |
|  Kategorie   | Prio | Kategorie editieren         | Status / Funktion |  |
|  HOME        | 1    | Kategorie editieren/löschen | online            |  |
|  NEWS        | 2    | Kategorie editieren/löschen | online            |  |
|  BOARDS      | 3    | Kategorie editieren/löschen | online            |  |
|  OPEN BOARD  | 4    | Kategorie editieren/löschen | online            |  |
|  USERLISTE   | 5    | Kategorie editieren/löschen | online            |  |
|              | 6    | Kategorie editieren/löschen | online            |  |
|  MEIN PROFIL | 7    | Kategorie editieren/löschen | online            |  |





Abb.: struktur.gif

## B1.02 Artikel

Über einen Artikel können Sie Inhalte einbinden und auf der Webseite erscheinen lassen. Wie werden nun Artikel in REDAXO erstellt und verwaltet?

### Erstellen

Über das (+)Symbol können Sie einen Artikel in der gewünschten Kategorie erstellen. Der Startartikel ist in jeder Kategorie grundsätzlich vorhanden und kann nicht gelöscht werden.

|  | Artikelname                                | Prio                           | Template  | Erstellt am   |                     |
|---|--|--------------------------------|---|---------------|---------------------|
|  | <input type="text" value="Neuer Artikel"/> | <input type="text" value="2"/> | default  | 21. Mar. 2006 | <b>Artikel</b>      |
|  | Sachsen                                    | 1                              | default   | 29. Nov. 2005 | <b>Startartikel</b> |

### Darstellung – Template

Jeder Artikel und dessen Inhalte wird in der Regel in ein Template, dem “Rahmen” der Webseite, geladen. Dazu wird beim Anlegen eines neuen Artikels das gewünschte Template aus der Templateliste ausgewählt. Daneben kann die Reihenfolge, mit der der Artikel z. B. in der Navigation aufgelistet wird, über die Einstellung **Prio** editiert und ein Artikelname angegeben werden. Über die Schaltfläche **add\_article** wird der Artikel mit dem ausgewählten Template gespeichert und angezeigt. Alle Angaben können nach dem Speichern bei Bedarf geändert werden.

### Funktion/Status – online/offline

Wie auch bei der Verwaltung der Kategorien, kann diese Funktion in der Programmierung genutzt werden. Die Navigation kann so festgelegt werden, dass Artikel erst angezeigt werden, wenn die Funktion auf “online” gesetzt wird. Für weitere Information zur Verwendung dieser Funktion erfahren Sie in dem Bereich [C. Entwickler & Admin](#)

| Status/Funktion |         |        |
|-----------------|---------|--------|
| ändern          | Löschen | online |
| ändern          | Löschen | online |
| ändern          | Löschen | online |

### Artikel – Inhalt

Über das Artikel-Icon oder den Artikelnamen können Sie zu der Verwaltung der Inhalte eines Artikel wechseln. Hier wird der Artikel im **Editiermodus** geöffnet, d.h. hier werden alle Inhalte in **Blöcken** dargestellt und verwaltet. Wechseln Sie zu **Anzeigen** und die Inhalte werden ohne die Darstellung in Blocks angezeigt.

The screenshot shows the REDAXO editor interface. At the top, there are navigation options: 'Typen: | mitte | rechts |', 'Anzeigen | Editier Modus | Metadaten/Sonstiges'. Below this is a search bar with the text 'Block hinzufügen'. The main content area displays a list of blocks:

- 01 - Headline** with buttons 'Editieren' (green) and 'Löschen' (red), and up/down arrows.
- Neuigkeiten in und über Sachsen** (the headline text)
- Another search bar with 'Block hinzufügen'.
- 01 - Text und/oder Bild [textile]** with buttons 'Editieren' (green) and 'Löschen' (red), and up/down arrows.
- The text: 'diese Seite verwendet das Modul "Teaser".'
- Another search bar with 'Block hinzufügen'.
- 05 - Teaser** with buttons 'Editieren' (green) and 'Löschen' (red), and up/down arrows.

Sie haben bei der Erstellung eines Artikels, die Möglichkeit, die Inhalte in einem oder mehreren Blöcken zu verwalten. Mit einer Auswahl an Modulen wie Headline, Text, Bild oder Links steht Ihnen eine flexible und freie Redaktion der Inhalte zur Verfügung. Sie können vor, zwischen und hinter jedem Block einen weiteren Block aus der Blockliste auswählen und einfügen.

Mehr zu Blöcken und Modulen erfahren Sie im nächsten Kapitel [B1.03 Block/Modul](#)

### Metadaten/Sonstiges

Die Eingaben und Einstellungen dieser Seite können in der Programmierung genutzt und für Anzeige und Verwendung auf der Webseite verwendet werden. Dies ist im Kapitel "C1.04 Seitenspezifische Metadaten" im Detail beschrieben.

## B1.03 Block/Modul

Die eigentlichen Inhalte werden aus Modulen zusammengebaut. Man könnte sie auch Minitemplates nennen, wobei eine unendliche Anzahl von diesen in einem Artikel möglich sind. Diese Module können sehr unterschiedlich sein. Mögliche Formen eines Moduls sind z.B. Headline, Fliesstext, Grafiken hochladen und darstellen oder dynamische Listen aus Datenbanken, dynamische Grafiken, Unternavigationen etc.

### Wie wird ein Modul/Block (Inhalt) einem Artikel zugefügt?

Öffnen Sie den Artikel, in dem Sie ihre inhaltlichen Änderungen hinzufügen wollen, über das Artikelicon oder den Artikelnamen. Wählen Sie aus der Modulliste ein Modul aus und editieren Sie nun nach den Angaben ihren Inhalt, z.b geben Sie einen Text in ein Feld ein.

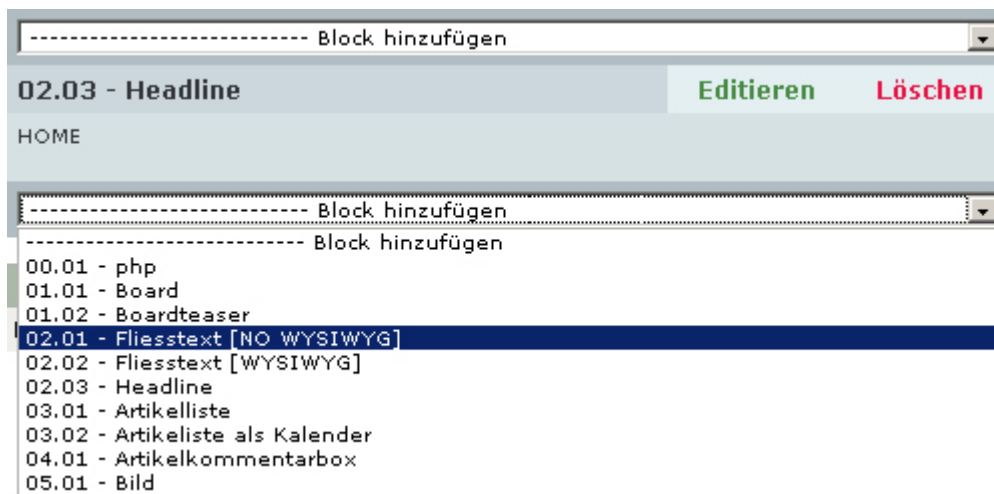


Abb.: article\_select\_modul.jpg

Über **Block hinzufügen** wird das Modul zu einem **Block** in dem Artikel. Der Artikel wird nun mit dem hinzugefügten Block erneut geladen. Über **Editieren** können Sie den Inhalt editieren und erneut speichern. Über **Löschen** können Sie den Block aus dem Artikel entfernen.

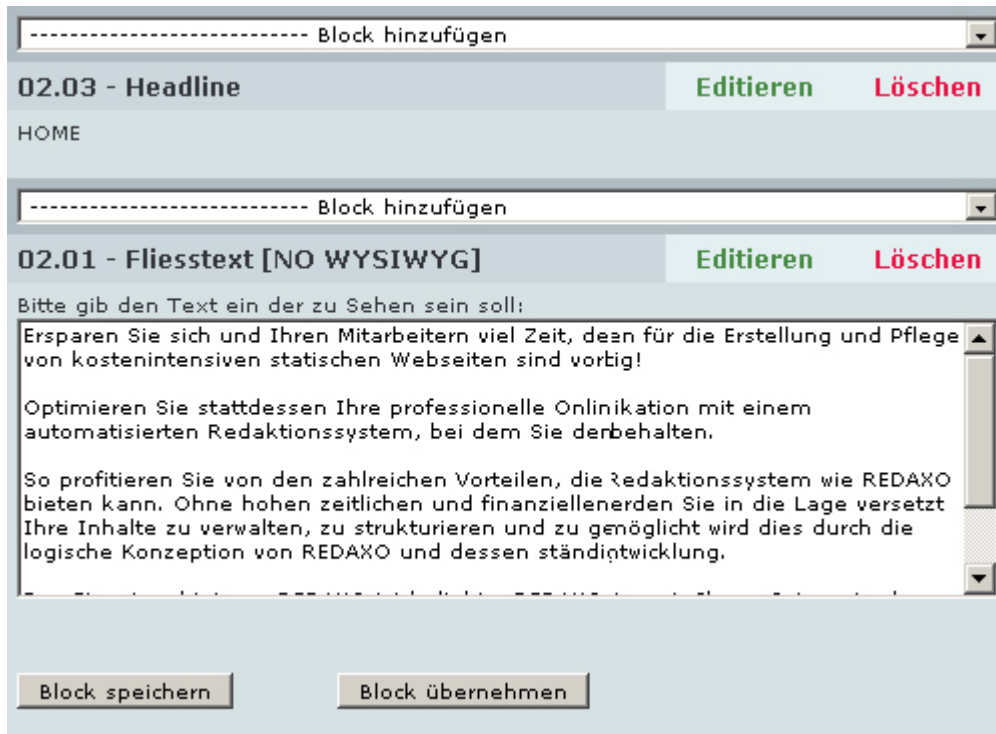


Abb.: modul\_addcontent.jpg

**Wie wird die Modulliste erstellt?**

Die Modulliste wird in der Regel vom Entwickler im Bereich **Module** erstellt. Im Bereich Module wird die Eingabe (Ansicht in der Redaktion) sowie die Ausgabe (Darstellung Webseite) vom Entwickler definiert und gestaltet.

Dies wird im Kapitel [C2. Module & Aktionen](#) genauer beschrieben.













| Module  |                                  |  |     |      |
|---|----------------------------------|---|-----|------|
| Modules   Actions   |                                  |   |     |      |
|  | Modulbezeichnung                 | Funktionen  | PHP | HTML |
|  | 00.01 - php                      | Modul löschen   |     |      |
|  | 01.01 - Board                    | Modul löschen   |     |      |
|  | 01.02 - Boardteaser              | Modul löschen   |     |      |
|  | 02.01 - Fliesstext [NO WYSIWYG]  | Modul löschen   |     |      |
|  | 02.02 - Fliesstext [WYSIWYG]     | Modul löschen   |     |      |
|  | 02.03 - Headline                 | Modul löschen   |     |      |
|  | 03.01 - Artikelliste             | Modul löschen   |     |      |
|  | 03.02 - Artikeliste als Kalender | Modul löschen   |     |      |
|  | 04.01 - Artikelkommentarbox      | Modul löschen   |     |      |
|  | 05.01 - Bild                     | Modul löschen   |     |      |

Abb.: modul\_list.jpg

Mittlerweile gibt es eine umfangreiche Sammlung von Modulen, die zum Download angeboten werden.

## B2. Medienpool

Über den Medienpool können Dateien auf den Server geladen, verwaltet und gelöscht werden. Im allgemeinen werden darüber Grafiken und Textdateien (z. B. Stylesheets) verwaltet. Es können aber auch beliebige andere Dateitypen bearbeitet werden.

Der **Medienpool** wird in einem Popup-Fenster aufgerufen.

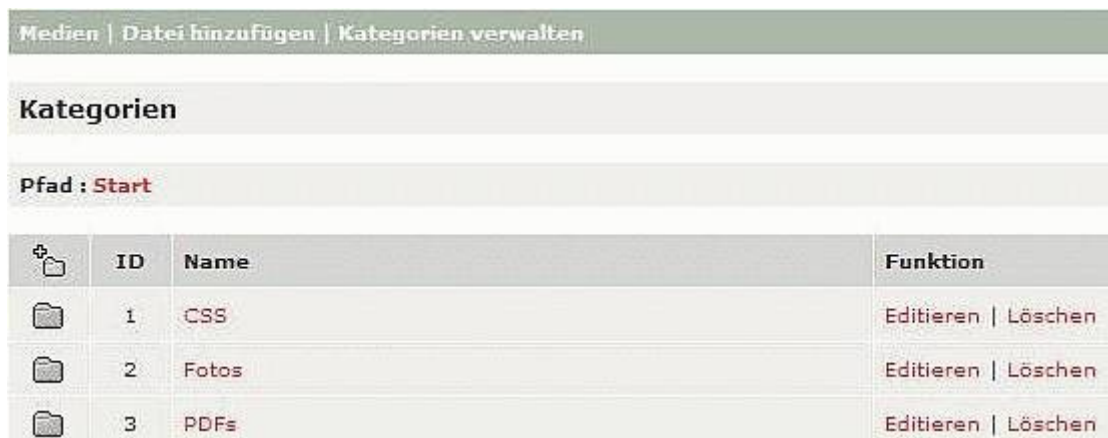
### Medien verwalten

Alle Dateien, die mit dem Medienpool verwaltet werden, befinden sich in dem Ordner "files". In dem Medienpool werden nur die Dateien angezeigt, die über den Medienpool in das Verzeichnis geladen wurden.

Um die Sortierung übersichtlicher zu gestalten, kann man Kategorien definieren und die hochgeladenen Dateien diesen Kategorien zuweisen. In der Demo-Version von Redaxo wird z. B. zwischen Bildern, Hilfsgrafiken und Stylesheets unterschieden. Es können beliebig viele Kategorien definiert werden.

### Kategorien verwalten

Unter der Überschrift "Kategorien verwalten" können neue Kategorien angelegt und vorhandene editiert oder gelöscht werden. Zum Anlegen einer neuen Kategorie klickt man auf das Kreuzsymbol neben der Bezeichnung "Name" und trägt den Kategorienamen in das entsprechende Textfeld ein. Das Editieren beschränkt sich auf eine Umbenennung der Kategorie. Kategorien können nur gelöscht werden, wenn ihnen keine Dateien mehr zugewiesen sind. Ansonsten erscheint ein Warnhinweis.







|  | ID | Name  | Funktion            |
|---|----|-------|---------------------|
|  | 1  | CSS   | Editieren   Löschen |
|  | 2  | Fotos | Editieren   Löschen |
|  | 3  | PDFs  | Editieren   Löschen |

Abb.: mp\_kat.jpg

### Dateien hinzufügen

Zum Hochladen von Dateien in den Medienpool bzw. das Verzeichnis "files", wählt man den Menüpunkt "Datei hinzufügen". Hier können dann zu der Datei ergänzende Informationen angegeben werden, wie z. B. Titel, Beschreibung, Copyright.

Medien | Datei hinzufügen | Kategorien verwalten

### Datei hinzufügen

|               |  |
|---------------|--|
| Titel:        | <input type="text" value="Startbild"/>   |
| Kategorie:    | <input type="text" value="Fotos"/>   |
| Beschreibung: | <input type="text"/>   |
| Copyright:    | <input type="text"/>   |
| Datei:        | <input type="text" value="datei.jpg"/> <input type="button" value="Durchsuchen..."/> |
|               | <input type="button" value="Hinzufügen"/>  |

Abb.: mp\_addfile.jpg

### Medien: Übersicht

Zu jeder Kategorie kann man sich die entsprechenden Dateien anzeigen lassen. Durch Klick auf den Bildtitel erhält man die Detailinformationen.

Medien | Datei hinzufügen | Kategorien verwalten

### Medien

| Kategorien               | <input type="text" value="Fotos"/> <input type="button" value="suchen"/>   |            |
|--------------------------|--|------------|
| Thumbnail                | Dateiinfo/ Beschreibung  | Funktionen |
| <input type="checkbox"/> | <b>Sächsische Schweiz</b><br><b>saechs_schweiz_01.jpg [10.62 KBytes]</b><br>[Keine Beschreibung eingegeben]<br>30-Nov-2005   02:36h   dergel |            |

Abb.: mp\_menue.jpg

### Medien – Detailansicht

Klicken Sie in der Übersicht einer Kategorie auf eine Datei und man gelangt zu Detailansicht. Über die Detailansicht können die Kategorie wechseln, die Datei ändern, Beschreibungs- und Copyrighttexte einfügen oder die Datei löschen bzw. aktualisieren. Über die Schaltfläche "Aktualisieren" werden die Änderungen übernommen.

Datei können aus dem Medienpool nur gelöscht werden, wenn die Datei nicht in einem Artikel/Block eingefügt ist.

### Medien: Detailansicht

|   |  |
|---|--|
| <b>Kategorien</b>   | Fotos <input type="button" value="suchen"/>                        |
| <b>Detailinformationen</b>  |  |
| Titel:  | Sächsische Schweiz   |
| Kategorie:  | Fotos <input type="button" value="↓"/>                             |
| Beschreibung:   | <input type="text"/>   |
| Copyright:  | <input type="text"/>   |
| Dateiname:  | saechs_schweiz_01.jpg  |
| Letzter Update:   | 30. Nov, 2005 - 02:36 [dergel]                                     |
| Erstellt am:  | 29. Nov, 2005 - 14:30 [thomas]                                     |
| Datei austauschen:  | <input type="text"/> <input type="button" value="Durchsuchen..."/> |
| <input type="button" value="Aktualisieren"/> <input type="button" value="Löschen"/> |  |



Copyright by

## **B3. Statistiken**

Redaxo bietet umfangreiche Statistikfunktionen, mit denen man das Besucherverhalten für einen Webauftritt analysieren kann.

Im Einzelnen werden ausgewertet und angezeigt:

### **Zeitliche Verteilung der Visits**

- Anzahl der Besuche gelistet nach Tagen oder Monaten,
- die Anzahl der Pageviews und
- die durchschnittliche Anzahl der Pageviews pro Besuch

### **Besuchte Artikel**

- Verteilung der Pageviews pro Artikel,
- die am häufigsten und
- die selten aufgerufenen Artikel

### **Aussagen über die Besucher**

- Herkunft der Besucher sortiert nach Land,
- die Häufigkeit von Suchmaschinenzugriffen,
- die Suchbegriffe, die zu der Seite geführt haben und die Referer-URLs der Besucher

### **Aussagen über die Browsertypen**

- Verteilung der Browser, mit denen auf die Seite aufgerufen wurde und
- die Verteilung der entsprechenden Betriebssysteme

### **Auswertung erstellen**

Um die Statistik zu erhalten, müssen die Daten zunächst ausgewertet werden. Dazu wählt man unter dem Punkt "Auswertung" den gewünschten Monat und bestätigt durch Anklicken des Buttons "Auswerten". Anschließend kann die neu erstellte oder aktualisierte Statistik unter dem Punkt "Statistik" angezeigt werden.

## C. Entwickler & Admin

In diesem Bereich finden Sie als Entwickler oder Administrator weiterführende Informationen zur Entwicklung und Programmierung einer Webseite.

Lernen Sie das Konzept von REDAXO kennen und erweitern so die Funktionen im Backend und auf Ihrer Webseite.

### C1. Templates erstellen

Mit Hilfe eines Templates wird das Layout einer Seite (eines Artikels) festgelegt. Hier wird die generelle Struktur des Artikels einschließlich der Navigation ...

### C2. Module & Aktionen

Mehr zur Entwicklung und Bereitstellung von Modulen und Aktionen ...

### C3. Benutzer

Über die Erstellung verschiedenen Benutzer-Account, regeln Sie auch die Rechtevergabe in REDAXO.

### C4. Addons

Addons erweitern den Funktionsumfang von REDAXO. Das Addon-Konzept erlaubt es REDAXO nach Ihren Wünschen zu erweitern ...

### C5. Specials

Auf dieser Seite stehen Informationen zur Verfügung, die im sich auf die Einstellungen in der master.inc.php beziehen. Daneben können Sie drei weitere Funktionen aktivieren ...

### C6. Object Oriented Framework

Sie können auf die Informationen in den Tabellen die REDAXO verwendet über SQL-Befehl zugreifen oder wie auf den folgenden Seite beschrieben, über ein Objekt-Orientiertes Framework.

### C7. Import & Export

Ein Import besteht in der Regel aus dem Import der Datenbank (Tabellen und Inhalte) und den Dateien, die beim Export erstellt wurden.

### C8. CVS

Im CVS können Sie aktuelle Entwicklungsversionen laden.

### C9. Extension Points

REDAXO bietet durch eine Extension Point API die Möglichkeit, via Addon/Modul/Template direkt in den Workflow einzugreifen.

# C1. Templates

Mit Hilfe eines Templates wird das Layout einer Seite (eines Artikels) festgelegt. Hier wird die generelle Struktur des Artikels einschließlich der Navigation definiert. Metatags, Stylesheets und Javascript können eingebunden werden.

Alle Artikel, denen ein bestimmtes Template zugewiesen wird, erhalten dadurch ein einheitliches Layout. Der artikelspezifische Inhalt wird jeweils über die Funktion `$this->getArticle()` eingefügt.

Es können Templates erstellt werden, die in den Artikeln selektiert werden sollen, oder welche, die in andere Templates eingebunden werden sollen.

## C1.01 Template erstellen

Im einfachsten Fall besteht ein Template aus einer leeren html-Seite und einer Referenz auf den aktuellen Artikel:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title></title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
<meta name="keywords" content="wort1, wort2, ...">
<link rel="stylesheet" type="text/css" href="css/style.css">
</head>

<body>

<?
/* hier wird der artikelspezifische Inhalt eingebunden;
$this verweist dabei auf den jeweils aktuellen Artikel */

echo $this->getArticle();

?>

</body>
</html>
```

Dieses Beispiel kann beliebig variiert werden. Das Layout wird in einem Template üblicherweise detaillierter definiert. So können hier unterschiedliche Seitenbereiche, z. B. Kopf- und Fußzeilen oder mehrere Spalten formatiert und die Navigation eingebunden werden. Der artikelspezifische Inhalt wird in einem beliebigen Bereich/Container eingefügt. Es ist auch möglich, die Inhalte mehrerer Artikel in einem Template anzuzeigen (s. Kap. C1.03).

## C1.02 Navigation

Es gibt unterschiedliche Möglichkeiten, eine Navigation festzulegen:

### Externe Links

#### Template

Links auf externe Seiten werden in einem Template in der üblichen Form festgelegt:

```
<a href=http://www.redaxo.de> Redaxo-Hompage </a>
```

#### Modul

In den Modulen hängt die Schreibweise für einen externen Link davon ab, wie die Ein- und Ausgabe definiert sind, welche Redaxo-Variablen verwendet werden und ob eine automatische Umwandlung bestimmter Signalwörter vorgesehen ist. Hier muß berücksichtigt werden, mit welchem Modul man aktuell arbeitet.

### Interne Links, manuell definiert

#### Template

Soll in einem Template auf interne Seiten verlinkt werden, wird die index-Seite als Adresse angegeben und die gewünschte Artikel-ID als Variable übergeben.

```
<a href=index.php?article_id=22> Interner Link Artikel 22</a>
```

Eine andere Möglichkeit stellt folgende Schreibweise dar:

```
<a href="< ?= getUrlbyID(22) "> Interner Link Artikel 22</a>
```

Dann funktionieren die Links auch wenn RexRewrite aktiv ist.

Siehe [Wiki: TextUrlsOhneModRewrite](#)

### Navigation - Beispiel:

#### **3 Kategorie Ebenen (kein Unterscheidung online/offline)**

Mit diesem Quellcode werden bis zu drei Ebenen von Kategorien aufgelistet. Es werden alle Kategorien und Unterkategorien angezeigt. Eine Unterscheidung je nach Status "online" oder "offline" wird hier nicht gemacht.

### Navigations-Generator

Für die Erstellung und Anpassung weitere Beispiele, wurde dieser "Generator" von user Mediastuttgart zur Verfügung gestellt. Das Script generiert automatisch bis zu 10 Ebenen inklusive CSS für die aktiv und inaktiv Klassen. Dabei sind alle CSS Klassen getrennt.

[http://www.mediastuttgart.de/ms\\_rex\\_gen.php](http://www.mediastuttgart.de/ms_rex_gen.php)

## Module

Bei Verwendung eines Moduls werden der Name und die ID eines Artikels als Formularwerte übergeben. Dem Formular wird der Name **Link[1]** zugewiesen.

Die Ausgabe erfolgt in dem Modul mittels der Redaxo-Variablen **REX\_VALUE[1]**

## Interne Links, dynamisch erzeugt

Statt einer manuell erzeugten Navigation kann man diese auch dynamisch erstellen lassen. Dazu bieten sich mehrere Möglichkeiten. Man kann sowohl das OO-Konzept von Redaxo nutzen als auch die Struktur mit Hilfe von SQL-Befehlen ermitteln und darstellen. Siehe dazu die Beispiele in den Folgekapiteln.

## Sitemap, dynamisch generieren

Nach dem gleichen Prinzip kann man eine Sitemap dynamisch generieren lassen. Das hat den Vorteil, dass die Sitemap bei Änderungen an der Kategorie- oder Artikelstruktur automatisch aktualisiert wird.

## Download von Navigationen

Im Downloadbereich findet man bei den Templates einige Beispiele für Navigationen.

s. [http://www.redaxo.de/19-0-templates.html?cat\\_id=2](http://www.redaxo.de/19-0-templates.html?cat_id=2)

### C1.02.01 Navigation

Mit diesem Quellcode werden bis zu drei Ebenen von Kategorien aufgelistet. Es werden alle Kategorien und Unterkategorien angezeigt. Eine Unterscheidung je nach Status "online" oder "offline" wird hier nicht gemacht.

Dies ist nur ein Beispiel, das den eigenen Anforderungen angepasst werden kann. Zur Erzeugung der Navigation werden Elemente des Objekt-Orientierten Frameworks genutzt:

#### OOCategory::getRootCategories() as \$cat

Es werden die Kategorien der ersten Ebene ausgelesen.

`$this->getUrl()`

Für die Verlinkung wird die URL des Startartikels, der zu dieser Kategorie gehört, wird ausgelesen.

`$this->getName()`

Es wird der Name der Kategorie ausgelesen.

`$this->getChildren()`

Für eine bestimmte Kategorie werden die Unterkategorien ausgelesen.

```
<ul>
<!-- top level categories -->
<?foreach (OOCategory::getRootCategories() as $cat):?>
<li><a href='<?=$cat->getUrl()?>'><?=$cat->getName()?></a><ul>

<!-- 1st level categories -->
<?foreach ($cat->getChildren() as $sub1):?>
<li><a href='<?=$sub1->getUrl()?>'><?=$sub1->getName()?></a><ul>

<!-- 2nd level categories -->
<?foreach ($sub1->getChildren() as $sub2):?>
<li><a href='<?=$sub2->getUrl()?>'><?=$sub2->getName()?></a></li>
<?endforeach;?>

</ul></li>
<?endforeach;?>

</ul></li>
<?endforeach;?>
</ul>
```

## C1.03 Mehrere Artikel einbinden

Der Inhalt des aktuell aufgerufenen Artikels wird durch Aufruf der Funktion `$this->getArticle()` eingefügt.

Es können darüber hinaus Inhalte beliebiger weiterer Artikel eingefügt werden. Dies geschieht, indem ein neuer Artikel deklariert und diesem eine bestimmte Artikel-ID zugewiesen wird. Die ID gehört zu dem Artikel, dessen Inhalt hier angezeigt werden soll. Er muß in der Struktur definiert sein. In diesem Artikel können Inhalte gespeichert werden, die auf mehreren Seiten angezeigt werden sollen, z. B. die Navigation oder eine News-Liste.

Der Inhalt wird ebenfalls durch Aufruf der Funktion `$artikel_2->getArticle` eingefügt.

Es ist möglich, einen zusätzlichen Artikel mehrfach in eine Seite einzubinden oder mehrere unterschiedliche Artikel innerhalb einer Seite einzubinden.

Das folgende Listing soll das das Prinzip erläutern. Das Einbinden in Tabellenfelder ist nur ein Beispiel.

```
<?
/* hier wird der Inhalt eingefügt, der zu dem aktuellen Artikel gehört */

echo "<table><tr><td>".$this->getArticle()."</td>";

/* hier wird ein weiterer Artikel eingebunden */

$artikel_2 = new article;
$artikel_2->setArticleID(123);

echo "<td>".$artikel_2->getArticle()."</td></tr></table>";
?>
```

## C1.04 Seitenspezifische Metadaten

Redaxo bietet die Möglichkeit, für jeden Artikel spezifische Metaangaben zu definieren und auszugeben. Dadurch können die Seiten z. B. von den Suchmaschinen gezielter gefunden werden.

Die Festlegung dieser Meta-Informationen erfolgt in der Kategorie "Struktur". Der gewünschte Artikel wird ausgewählt und die Bearbeitungsmaske "Metadaten/Sonstiges" aufgerufen.

### Online vom - bis zum

Es besteht die Möglichkeit, hier einen Zeitrahmen vorzugeben, innerhalb dessen der Artikel online sein soll.

### Seitenspezifische Metatags

Die Felder "Name/Bezeichnung", "Suchbegriffe" und "Beschreibung" können genutzt werden, um seitenspezifische Metatags in einen Artikel einzufügen.

Zum Aufrufen dieser Informationen werden in dem Template folgende Befehle eingefügt:

```
// Titel
<title><? echo $this->getValue("name"); ?></title>
```

```
// Beschreibung
<meta name="description" content="<? echo $this->getValue("description"); ?>" />
```

```
// Suchworte
<meta name="keywords" content="<? echo $this->getValue("keywords"); ?>" />
```

### Seitenspezifische Bilder

Dem Artikel kann hier ein spezielles Bild zugewiesen werden. Der Aufruf dieses Bildes zum Artikel erfolgt über den Befehl

```
$this->getValue("file")
```

So kann man das Bild in einem Template/Modul anzeigen lassen:

```
">
```

### Teaser

Der Artikel kann durch Selektion des Feldes "Teaser" als Teaser gekennzeichnet werden. In der Datenbank wird in der Tabelle rex\_article der Wert für die Spalte "Teaser" auf 1 gesetzt und kann so über ein entsprechendes Modul selektiert werden.

Den Wert erhält man über den Befehl

```
$var = $this->getValue("teaser");
```

### Artikeltyp

Sind unterschiedliche Artikeltypen definiert, kann hier eine Auswahl und Zuweisung zum Artikel erfolgen.

### Sonstige Funktionen

In Abhängigkeit von den Berechtigungen eines Benutzers, erhält dieser weitere Funktionen angezeigt.

| Sonstige Funktionen   |  |
|-----------------------|--|
| Inhalte von Sprache   | deutsch <input type="button" value="v"/> nach <input type="text" value="kisuaheli"/> <input type="button" value="v"/> kopieren |
|                       | <input type="button" value="Inhalte kopieren"/>  |
| Artikel kopieren      | Über uns   |
|                       | <input type="button" value="Artikel kopieren"/>  |
| Kategorie verschieben | Über uns   |
|                       | <input type="button" value="Kategorie verschieben"/>   |

**"copyContent" - ab Version 3.0**

Hat der Benutzer die Berechtigung "copyContent" und sind mehrere Sprachen eingerichtet, so bekommt er hier die Möglichkeit angezeigt, Inhalte eines Artikels von einer Sprache in eine andere zu kopieren.

**copyArticle - ab Version 3.2**

Benutzer mit der Berechtigung "copyArticle" haben hier die Möglichkeit einen Artikel in eine andere Kategorie zu kopieren. Die Kategorien werden hier zur Auswahl angezeigt.

**moveCategory (nur Startartikel) - ab Version 3.2**

Benutzer mit der Berechtigung moveCategory können komplette Kategorien verschieben. Dies geschieht durch Verschieben des Startartikels dieser Kategorie. Die Kategorie wird dann mit allen Artikeln und Unterkategorien verschoben.

**moveArticle ("normale" Artikel) - ab Version 3.2**

Artikel, die keine Startartikel sind, können einzeln in eine andere Kategorie verschoben werden. Hier wird dann statt "Kategorie verschieben" die Option "Artikel verschieben" angezeigt.

|                     |  |
|---------------------|--|
| Artikel verschieben | Home   |
|                     | <input type="button" value="Artikel verschieben"/> |
| Artikel kopieren    | Home   |
|                     | <input type="button" value="Artikel kopieren"/>    |

## C1.05 Includes

Es besteht die Möglichkeit, Programmcode in einer separaten Text-Datei auszulagern und diesen dann mittels include()-Befehl an einer bestimmten Stelle in ein Template einzufügen.

Die include-Datei kann beliebigen php-Code enthalten. Sinnvollerweise sollte sie in dem für includes vorgesehen Ordner redaxo/include gespeichert werden. Wenn sie in einem anderen Ordner abgelegt wird, muß bei dem include-Befehl die Pfadangabe entsprechend angepasst werden. Nach einer allgemeinen Übereinkunft werden include-Dateien mit der Endung .inc.php gespeichert.

Ein Beispiel für den Einsatz von includes ist das Auslagern einer Navigation in einem Template. Durch die separate Ablage des Codes wird das Template übersichtlicher. Das Layout kann dadurch leichter bearbeitet werden.

Ein einfaches Beispiel:

```
<div>
<?
// hier wird der ausgelagerte Code eingelesen\
include($REX[INCLUDE_PATH].'/navigation.inc.php');
?>
</div>
```

```
<div>
<?
// hier wird der Seiten-Inhalt eingelesen
$this->getArticle();
?>
</div>
```

Je nach Erfordernis kann nach diesem Prinzip Programmcode ausgelagert und wieder eingelesen werden.

Wie in den Demos gezeigt, ist es auch möglich, andere Templates, die Sie in Redaxo erstellt haben, einzulesen. In diesem Beispiel wird ein bereits generiertes Template eingelesen. Infos zu generiertes Dateien in REDAXO ...

```
<?
//Einlesen eines weiteren Templates
include $REX[INCLUDE_PATH]."/generated/templates/2.template";
?>
```

## C1.06 CTYPES

(Verfügbar ab Version 3.1)

Mit der Version 3.1 wurde ein neues Konzept eingeführt, in einem Template mehrere Bereiche mittels sogenannter 'CTYPES' zu definieren und diese artikelspezifisch mit Inhalt zu füllen. Dadurch ist es jetzt einfacher mehrspaltige Layouts oder spezielle Kopf- und Fußbereiche zu realisieren. Es wird empfohlen die ctypes einzusetzen, wenn ein kompletter Internetauftritt auf einem Template basiert. Bei Einsatz unterschiedlicher Templates mit mehreren Bereichen kann man ansonsten sehr schnell den Überblick verlieren.

Die Bereiche werden in der Datei ...redaxo\include\ctype.inc.php festgelegt. Zur Zeit muß diese Datei noch mittels eines Editors bearbeitet werden. Eine Bearbeitung über das Redaxo-Backend ist jedoch für Folgeversionen geplant.

Die „Platzhalter“ der einzelnen Bereiche werden in der Variablen \$REX['Ctype'][x] gespeichert. Theoretisch ist es möglich, beliebig viele ctypes zu definieren. Praktisch ist das aber weder nötig noch sinnvoll.

### Beispiel:

Ein Template mit drei Bereichen (Rechte Spalte, Linke Spalte, Fußbereich)

Angaben in der ctype.inc.php:

```
// --- DYN CTYPE

$REX['CTYPE'][0] = "main";
$REX['CTYPE'][1] = "links";
$REX['CTYPE'][2] = "unten";

// --- /DYN
```

Die Benennung der einzelnen ctypes ist beliebig. Die Namen erscheinen im Editier-Modus zu den Artikeln. In den Templates werden die ctypes mittels der Array-Laufzahl angesprochen.

Die Referenzierung der ctypes im Template könnte z. B. so aussehen:

```
<div id="rechte-spalte">
<?php
print $this->getArticle(0);
?>
</div>
```

```
<div id="linke-spalte">
<?php
print $this->getArticle(1);
?>
</div>
```

```
<div id="fusszeile">
<?php
print $this->getArticle(2);
?>
</div>
```

Die div-Tags werden hier nur beispielhaft eingesetzt. Die Einbindung der ctypes kann in jedem

beliebigen, sinnvollen html-Kontext erfolgen.

Wenn man jetzt im Backend einen Artikel mit Inhalt versehen möchte, hat man im Editier-Modus die Möglichkeit, jedem definierten Bereich Blöcke zuzuweisen. Unter „Typen“ werden die Namen angezeigt, die man den Bereichen in der ctype.inc.php zugewiesen hat.

Bereich 'main':

Bereich 'links':

Die Inhalte der unterschiedlichen Bereiche werden für diesen Artikel mit der ctype-ID des jeweiligen Bereiches gespeichert. Beim Aufruf des Artikels werden die Bereiche „main“, „links“ und „unten“ mit den artikelspezifischen Inhalten angezeigt.

## C2. Module & Aktionen

Bevor Sie sich mit der Entwicklung und Bereitstellung von Modulen und Aktionen beschäftigen, sollten Sie sich mit dem Konzept, der Verwendung und sowie Verarbeitung der Inhalte über Module und Blöcke vertraut machen. ([B1.03 Block/Modul](#))

### Module

Im Bereich Module öffnet sich eine Liste mit allen Modulen die z.B. in der installierten Demo vorhanden sind. Über (+)-Symbol können Sie eigene Module erstellen und so für die Redaktion der Inhalte, in den Artikeln zur Verfügung stellen.

### Eingabe und Ausgabe

Öffnen oder erstellen Sie ein neues Modul und Sie erkennen das einfach Prinip eines Moduls. Die Eingabe definiert dabei die Eingabemaske, die Sie als Redakteur sehen, wenn Sie einen Block hinzufügen oder über Editieren bearbeiten wollen. In der Ausgabe können Sie Ausgabe oder Darstellung auf der Webseite, der eingegebenen Inhalte definieren.

In beiden Felder können Sie die Anzeigen über HTML und PHP erstellen.

Für die Speicherung der verschiedenen Inhalte stehen Ihnen in REDAXO eigene Variablen zur Verfügung.

### Aktionen

Aktionen werden in Verbindung mit Modulen genutzt. Mit einer Aktion können die in einem Modul eingegebenen Variablenwerte bearbeitet werden. Es sind beliebige, in PHP codierte Operationen möglich. Wollen Sie z.B. beim Editieren eines Blocks, die Eingabe eines Textfeldes überprüfen, dann können Sie dies mit Hilfe einer Aktion tun. Die Eingabe kann, je nach Definition und Programmierung der Aktion, überprüft, korrigiert und verändert werden.

Die Definition und Funktionen einer Aktion erfolgt in einer einfachen Eingabemaske. Stehen Aktionen zur Auswahl, so wird bei der Erstellung oder Bearbeitung eines Moduls, diese Auswahl angezeigt kann dem Modul hinzugefügt werden.

### Backend: Blöcke – Slices

Die über ein Modul eingegeben Inhalte werden in der Tabelle "rex\_article\_slice" gespeichert. Die Zuordnung zu einem Artikel, wird über das Feld "article\_id" autom. eingetragen. Die Reihenfolge wird über das Feld "re\_article\_slice\_id" geregelt. Dabei wird in diesem Feld, beim Hinzufügen eines neuen Blocks im Artikel, der darüber stehende Block mit dem Wert aus dem Feld "ID" eingetragen. Die Slices bilden so eine Kette. Bei der Ausgabe wird die Kette aufgelöst und als Seite dargestellt.

Über das Feld "article\_id" können somit z.B. einzelnen Slices ausgelesen und dargestellt werden.

## C2.01 Modul erstellen und bereitstellen

Bei der Erstellung eines Moduls, haben Sie Möglichkeit, zwei Einteilungen zu verwenden.

### Eingabe

Einerseits das Eingabeformular eines Moduls. Mit diesem kann die Eingabemaske erstellt werden, die beim Editieren oder Hinzufügen einen Blocks im Artikel erscheint.

### Ausgabe

Andererseits das Ausgabeformular eines Moduls. Mit diesem kann die Ausgabe der eingegebenen Informationen angezeigt werden.

Eine einfache Form eines Modules:

Eingabe:

```
<input type=text size=20 name=VALUE[1] value="REX_VALUE[1]">
```

Ausgabe:

```
<font size=3>REX_VALUE[1]</font>
```

Dieses Modul erfaßt einen Text, welcher dann formatiert ausgegeben wird. Allgemein werden bestimmte Platzhalter verwendet um unterschiedliche Texte, Grafiken etc. dann dafür wieder einsetzen zu können. Hier wird im der Platzhalter REX\_VALUE-[1]- verwendet.

Ein einfaches Modul zum Bildupload mit Bildbeschreibung:

### Eingabe:

```
<?
echo "Bild: <br>";
if ("FILE[1]" != "")
{
echo "<img src=../files/FILE[1] width=100><br>";
echo "<input type=checkbox name=FILEDEL1> delete";
}else
{
echo "<INPUT NAME=FILE1 TYPE=file size=2>";
}
?><br><br>Bildunterschrift:<br>
<input type=text size=50 name=VALUE[1] value="REX_VALUE[1]" class=inp100>
<br><br>
```

### Ausgabe:

```
<?
if ("FILE[1]" != "") echo "<img src=/files/FILE[1]>";
if ("REX_VALUE[1]" != "") echo "<br>REX_VALUE[1]";
echo "<br><br>";
?>
```

In diesem Modul hat der Redakteur die Möglichkeit ein Bild hochzuladen und die Bildunterschrift einzugeben. Dabei werden folgende Platzhalter verwendet. FILE-[1]-, REX\_VALUE-[1]- sowie feste Formularwerte wie VALUE-[1]-, FILE1, FILEDEL1. Sie können innerhalb der Modul HTML oder PHP verwenden.

## C2.02 REDAXO Variablen

### REX\_VALUE[..] | VALUE[..]

Enthält Text-Eingaben, bis zu 10 Felder (REX\_VALUE[1]..) möglich.

Beispiel

Modul-Eingabe:

```
<input type=text size=20 name=VALUE[1] value="REX_VALUE[1]">
```

Modul-Ausgabe:

```
<font size=3>REX_VALUE[1]</font>
```

### REX\_HTML\_VALUE[..]

...

Beispiel

Modul-Eingabe:

```
<input type=text size=20 name=VALUE[1] value="REX_VALUE[1]">
```

Modul-Ausgabe:

```
<font size=3>REX_HTML_VALUE[1]</font>
```

### REX\_LINK[..] | LINK[..]

Bis zu 10 Felder (REX\_LINK[1]..REX\_LINK[10]) möglich.

Enthält im folgenden Beispiel, im Feld "link1" des jeweiligen Slices die Artikel-Id. In der Ausgabe wird die ID als "index.php?article\_id=ID" ausgegeben.

Modul-Eingabe:

```
<input type=text size=30 name=VALUE[1] value="REX_VALUE[1]"><br>
<input type=text size=30 name=LINK[1] value="REX_LINK[1]">
```

Modul-Ausgabe:

```
<a href=REX_LINK[1]>REX_VALUE[1]</a>
```

### REX\_FILE[..] | FILE[..]

Bis zu 10 Felder möglich(..)

Enthält Dateinamen, die wie in diesem Beispiel, über den Medienpool geladen werden. Alle Dateien die über den Medienpool geladen werden, werden im Ordner "/files" gespeichert.

Modul-Eingabe:

```
REX_MEDIA_BUTTON[1]
```

Modul-Ausgabe:

```
<?
if ("FILE[1]" != "")
{
echo "<img src=$REX[HTDOCS_PATH]/files/FILE[1] width=100>";
}
?>
```

### **REX\_IS\_VALUE[1] ... REX\_IS\_VALUE[10]**

#### **REX\_HTML\_BR\_VALUE[1] . REX\_HTML\_BR\_VALUE[10]**

Der eingegeben Text wird genauso ausgegeben. Jeder Umbruch wird durch ersetzt. Um zu sichern das kein php-code enthalten ist wird < ? Und ? > gelöscht.

#### **REX\_PHP\_VALUE[1] . REX\_PHP\_VALUE[10]**

Der eingegeben Text wird genauso ausgegeben. Sofern php code eingegeben wurde wird dieser auch ausgeführt.

#### **REX\_ARTICLE\_ID**

Hier wird die aktuelle Artikel ID eingesetzt.

#### **REX\_CATEGORY\_ID**

Hier wird die aktuelle Kategorie ID eingesetzt.

#### **REX\_PHP**

Sofern PHP als Modul verwendet werden soll. Ist es am sinnvollsten diesen Platzhalter zu verwenden.

## C2.03 Actions

Aktionen können in Verbindung mit Modulen angewendet werden. Im Bereich Module/Aktionen können Aktionen erstellt, editiert und verwaltet werden. Die Aktionen können dann im jeweiligen Modul ausgewählt und dem Modul hinzugefügt werden.

Wechseln Sie über die Subnavigation zu Actions und Sie erhalten eine Auflistung aller zur Verfügung stehenden Aktionen.

| Struktur   Medienpool   Templates   Module   Benutzer   AddOn   Specials   Community   Statistiken   Import/Export |                                    |                |
|--|------------------------------------|----------------|
| Module: Actions  |                                    |                |
| Modules   Actions  |                                    |                |
| ☰  | Aktionsname                        | Funktionen     |
| ☰  | preg_replace - pre/add [PRE ADD]   | Aktion löschen |
| ☰  | preg_replace - pre/edit [PRE EDIT] | Aktion löschen |

pergopa kristinus gbr | redaxo.de | ?

Es gibt mehrere Möglichkeiten, Aktionen ausführen zu lassen:

**PRE:** Die Aktion wird ausgeführt, bevor ein Block gespeichert wird.

**POST:** Post-Aktionen dienen nicht zur Werteausgabe sondern nur fuer "sonstige" Aktionen nachdem der Block aktualisiert wurde. Post actions sind eher unwichtig. Pre actions sind weitreichender.

Es gibt drei unterschiedliche Stati:

**ADD:** Die Aktion wird bei der Ersterstellung eines Blocks ausgeführt.

**EDIT:** Die Aktion wird jedesmal ausgeführt, wenn der Block bearbeitet wird.

**DELETE:** Die Aktion wird ausgeführt, wenn der Block gelöscht wird.

### 1. Beispiel: Einfache Filterung einer Eingabe in einem Block (PRE, EDIT).

Beim Editieren eines Blocks soll die Eingabe gefiltert und manipuliert werden.

Modul-Eingabe:

```
<input type=field class=inp100 name=VALUE[1] value="REX_VALUE[1]">
```

Modul-Ausgabe:

```
REX_VALUE[1]
```

Über ein Aktion kann der Wert des Feldes VALUE-[1]- gelesen und verändert werden. Soll dies beim Editieren des Block in einem Artikel passieren, dann wird die Aktion mit PRE/POST: PRE und STATUS: Edit erstellt.

Eingabe-Aktion:

Über \$REX\_ACTION können alle Felder (z.B. [VALUE][1], [FILE][1], [LINK][1]) gelesen und neu gesetzt werden.

Im Modul muss die Aktion nun hinzugefügt werden. Wird das Modul als Block einem Artikel hinzugefügt, wird die Eingabe beim Editieren des Blocks überprüft und ggfs. verändert gespeichert. Soll die Überprüfung auch beim Hinzufügen des Moduls erfolgen, dann muss eine zweite Aktion mit gleichem Code erstellt und mit STATUS: Add dem Modul hinzugefügt werden.

## 2. Beispiel: Eingabe in verschiedenen Felder oder Tabelle speichern (PRE, ADD/EDIT)

Die Eingabe in Textfeld soll z.B in der Beschreibung eines Artikels (Metadaten) zusätzlich gespeichert werden.

Modul-Eingabe:

```
<textarea cols="80" rows="10" name="VALUE[4]" class="inp100"></textarea>
```

Mit der folgenden Aktion wird der Wert aus VALUE-[4]- in das Feld "Beschreibung" des Artikels geschrieben. Der Wert dieses Feldes wird mit jedem Artikel in der Tabelle "rex\_article" gespeichert.

```
<?
$ins = new sql;
$ins->query("update rex_article set
beschreibung='". $REX_ACTION[VALUE][4]."' where id='REX_ ARTICLE_ID'")
?>
```

Der STATUS muss in zwei Aktionen auf Edit und Add dem Modul hinzugefügt werden.

---

### 3. Beispiel: Die Variableneingabe auf über 10 Werte erweitern

```
<?
$newname = "";

for ($c=0;$c<99;$c++) {
$newname .= $rexname[$c]."~~";
}

if ($REX_ACTION[VALUE][1] != "") {
$rexname = split("~~",$REX_ACTION[VALUE][1]);
}

else {
$REX_ACTION[VALUE][1] = $newname;
}
?>
```

Dieser Code wird zweimal als Aktion angelegt. Einmal als POST / ADD + EDIT Aktion und einmal als PRE / ADD + EDIT Aktion.

Danach kann man die beiden Aktion an das gewünschte Modul anfügen. Mittels der Variablen rexname1 bis rexname99 können die Werte abgefragt werden.

### Aktionen - Download

Im Downloadbereich können bereits erstellte Aktionen heruntergeladen werden.

### C3. Benutzer

Über die Erstellung verschiedenen Benutzer-Account, regeln Sie auch die Rechtevergabe in REDAXO. Mit der Konfiguration des Account geben Sie Programmteile und Funktionen in REDAXO frei und können so auch sensible Bereiche verstecken und sperren.

Beispiele für die Rechtevergabe können sein:

#### REDAXO-Navigation

So können z.B. die Bereiche Templates und Module für einen Grafiker gesperrt und der Bereich Medienpool frei geschaltet werden.

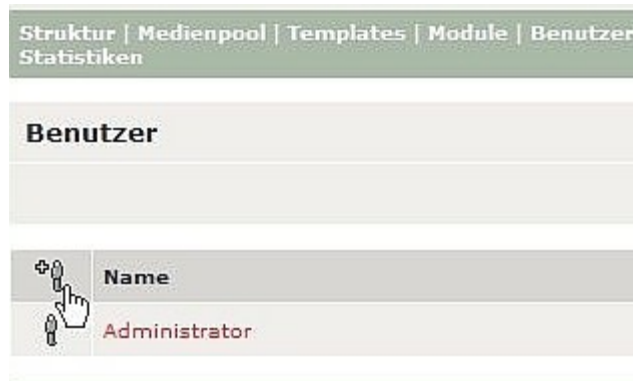
#### Kategorien und Artikel

Sie können innerhalb der Strukturverwaltung die gewünschten Kategorien und Artikel über jeweilige ID freigegeben oder andere Seiten sperren.

#### Module

Es kann die Benutzung von einzelnen Modulen freigegeben oder gesperrt werden.

Es können beliebig viele Nutzer angelegt werden. Der Administrator wird beim Redaxo-Setup automatisch angelegt.



Um einen neuen Benutzer anzulegen, klickt man auf das Symbol oben links in der Rubrik 'Benutzer'. Dann bekommt man eine Maske angezeigt, die sämtliche Optionen enthält, die man einem Benutzer zuweisen kann.

| Benutzer bearbeiten               |                                     |              |                              |
|-----------------------------------|-------------------------------------|--------------|------------------------------|
| Benutzername                      | willi                               | Passwort     | willi123                     |
| Name                              | Willi Wichtig                       | Beschreibung | Dies ist nur ein Test-Nutzer |
| <input type="checkbox"/>          | Admin (Alle Rechte/ Alles sichtbar) |              |                              |
| Sprachenzugriff                   | deutsch                             |              |                              |
| Bitte Strg drücken bei Änderungen |                                     |              |                              |

Hier werden zunächst Name und Passwort festgelegt. Durch Setzen eines Häkchens bei 'Admin (Alle

Rechte / Alles sichtbar) kann man pauschal alle Rechte an diesen Nutzer vergeben. Ansonsten können einzelne Rechte durch Selektieren der Listenpunkte mit gedrückter Strg-Taste (WINDOWS) gezielt ausgewählt werden.

Ist eine Seite in mehreren Sprachen angelegt, kann der Zugriff auf einzelne Sprachen beschränkt werden.

Die Festlegung von Berechtigungen wird im Detail auf der Seite 'Rechtevergabe' beschrieben.

### C3.01 Rechtevergabe Version 3.0, 3.1

Diese Rechte können einem Benutzer zugesprochen werden.



Der Bereich 'Allgemein' regelt den Zugriff auf die Themen im Redaxo-Backend. Hier wird u. a. festgelegt, ob dieser Nutzer Templates erstellen oder verändern darf, oder ob er die Statistik einsehen darf.



#### advancedMode

Man kann die Metadaten zu den Artikeln sehen und bearbeiten.

#### moveslice

Damit kann ein Benutzer Blöcke/slices im Artikel verschieben.

#### copyContent

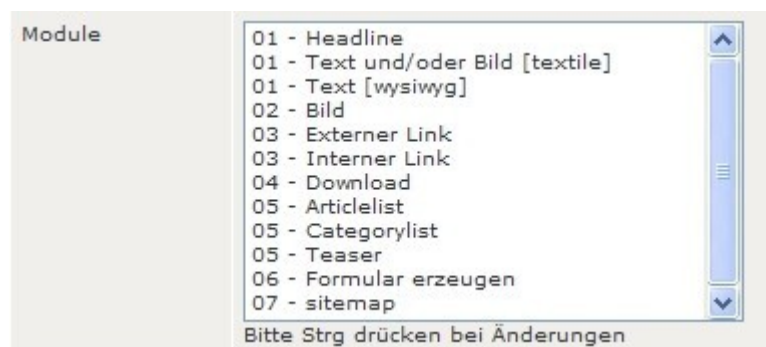
Der Inhalt eines Artikels kann kopiert werden.



Durch Anklicken der Checkbox bei 'Alle Kategorien' kann man pauschalen Zugriff auf alle Kategorien und Artikel gewähren. Ansonsten kann man einzelne Kategorien selektieren.



Es kann ein Zugriff auf alle oder auf einzelne Medienordner gewährt werden.



Hier wird festgelegt, mit welchen Modulen der Benutzer die Inhalte bearbeiten darf.

Das Feld **Extras** dient für alle Rechte, die man per Addon/Modul/Template einbauen und verwenden kann. Falls Extras festgelegt wurden, kann hier der Zugriff darauf gewährt werden.

## C3.02 Rechtevergabe Version 3.2

Diese Rechte können einem Benutzer zugesprochen werden.



Der Bereich 'Allgemein' regelt den Zugriff auf die Themen im Redaxo-Backend. Hier wird u. a. festgelegt, ob dieser Nutzer

- Zugriff erhält auf die Bereiche Addon, Specials und den Medienpool,
- Templates oder Module erstellen oder verändern darf,
- ob er die Statistik einsehen darf (wenn sie installiert und aktiviert wurde),
- ob er andere Benutzer anlegen und verändern darf,
- ob er die import/export-Funktionen nutzen darf.



`advancedMode[ ]`

Benutzer mit dieser Berechtigung dürfen die Metadaten zu den Artikeln sehen und bearbeiten.

`moveslice[ ]`

Benutzer mit dieser Berechtigung dürfen Blöcke/Slices im Artikel verschieben.

`copyContent[ ]`

Benutzer mit dieser Berechtigung dürfen den Inhalt eines Artikels von einer Sprachversion in eine andere Sprache kopieren. Ist diese Berechtigung aktiviert, wird die Option zum Kopieren bei den Metadaten angezeigt.

`moveArticle[ ]`

Benutzer mit dieser Berechtigung dürfen "normale" Artikel in eine andere Kategorie verschieben. Ist diese Berechtigung aktiviert, wird die Option zum Verschieben bei den Metadaten angezeigt. Dies erfolgt nur bei normalen Artikeln. Für Startartikel gilt die Entsprechung "moveCategory".

### copyArticle[ ]

Benutzer mit dieser Berechtigung dürfen Artikel in eine andere Kategorie kopieren. Ist diese Berechtigung aktiviert, wird die Option zum Kopieren bei den Metadaten angezeigt.

### moveCategory[ ]

Benutzer mit dieser Berechtigung dürfen durch Verschieben eines Startartikels die komplette Kategorie in andere Kategorien verschieben. Diese Option wird bei den Metadaten nur bei Startartikeln angezeigt.

### publishArticle[ ]

Benutzer mit dieser Berechtigung dürfen Artikel online/offline stellen.

### publishCategory[ ]

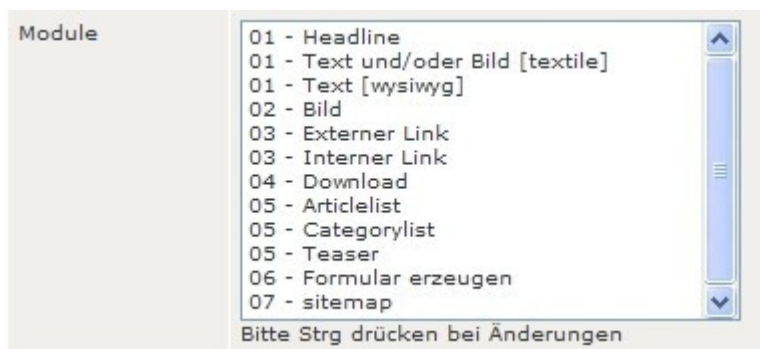
Benutzer mit dieser Berechtigung dürfen Kategorien online/offline stellen.



Durch Anklicken der Checkbox bei 'Alle Kategorien' kann man pauschalen Zugriff auf alle Kategorien und Artikel gewähren. Ansonsten kann man einzelne Kategorien selektieren.



Es kann ein Zugriff auf alle oder auf einzelne Medienordner gewährt werden.



Hier wird festgelegt, mit welchen Modulen der Benutzer die Inhalte bearbeiten darf.



Das Feld **Extras** dient für alle Rechte, die man per Addon/Modul/Template einbauen und verwenden kann. Falls Extras festgelegt wurden, kann hier der Zugriff darauf gewährt werden.

editContentOnly[ ]

Benutzer mit dieser "Berechtigung", die eigentlich eine Einschränkung ist, dürfen nur Inhalte verändern, nicht aber Änderungen an der Struktur vornehmen.

## C4. Addons

Addons erweitern den Funktionsumfang von REDAXO. Das Addon-Konzept erlaubt

- das Installieren von weiteren Navigationspunkten im Backend
- das Installieren von weiteren MySQL-Tabellen
- das Anlegen von Sprachversionen für das aktiviert Addon
- das Anlegen von Rechten für das Addon (über Benutzer)
- Manipulation des Workflows via Extension Points

### Addon: Import/Export

Mit der Installation von REDAXO erhalten Sie das ersten Addon. Es ist **installiert** und **aktiviert**. Die beiden Funktionen machen ein Addon im Backend verfügbar. [Weitere Addons](#) sind bereits geschrieben und können neben den eigenen Addons in REDAXO geladen werden.

### Addon-Installation

Für das Setup laden Sie die erforderlichen Dateien in das Verzeichnis "redaxo/include/addon/". Danach sollte im Bereich Addon das neue Addon verfügbar sein. Klicken Sie danach **Installieren** und danach **aktivieren**. Nach der Meldung "Addon aktiviert" sollte das Addon in der Navigation von REDAXO erscheinen. Für weitere Hinweise steht für jedes Addon eine Hilfedatei zur Verfügung und sollte vom Entwickler genutzt werden.

## C4.01 Struktur

Die Ordnerstruktur für ein Addon sieht in der Regel so aus:

```
/lang
/pages
config.inc.php
help.inc.php
install.inc.php
```

### config.inc.php

Konfigurieren Sie Ihr Addon in dieser Datei. Hier werden Einstellungen für die Sprachdateien, ID und Name, sowie die verwendeten Rechte geschrieben.

```
<?php
$mypage = "import_export"; // only for this file

// CREATE LANG OBJ FOR THIS ADDON
$I18N_ADDON = new
i18n($REX[LANG], $REX[INCLUDE_PATH]."/addons/$mypage/lang/");

$REX[ADDON][rxid][$mypage] = "REX_001"; // unique id /
$REX[ADDON][page][$mypage] = "$mypage"; // pagename/foldername
$REX[ADDON][name][$mypage] = "Import/Export"; // name
$REX[ADDON][perm][$mypage] = "import[]"; // permission
?>
```

Die Instanz für die Sprachversion Ihres Addons wird bei der Programmierung der Seiten verwendet.  
Hier: \$I18N\_ADDON

### help.inc.php

In diese Datei können Sie eine Beschreibung des Addons, Hilfen und Tipps, sowie Version und Autor in HTML beschreiben. Diese Seite erscheint, wenn auf der Seite "Addon" das (?) eines Addons angeklickt wird.

```
<b>Import Export</b>
<br><br>
```

### install.inc.php

Über diese Datei können Sie alle nötigen Schritte durchführen, die für die Installation des Addon benötigt werden. Dazu können Module installiert, Ordner angelegt, mySQL-Tabellen erstellt oder Seiten regeneriert werden (.).

```
<?php
// CREATE/UPDATE DATABASE
// CREATE/UPDATE MODULES
// CREATE/UPDATE PAGES
// CREATE/UPDATE FILES
// REGENERATE SITE

$REX[ADDON][install]["import_export"] = 1;
?>
```

### Sprachdateien (/lang)

Über die Sprachdateien können Sie Ihr Addon für die verschiedenen Sprachversionen im Backend anbieten. (Specials)

In dieser Datei können auf den Seite(n) des Addon verwendeten Elementen (Titel, Funktion, Bezeichnungen) definiert werden.

```
# addon:import/export en_GB
import = Import
importexport = Im-/Export
file_deleted = Datei wurde gelöscht
```

So kann auf einer Seite die Sprachdefinition genutzt werden:

```
<th align=left width=50%><?php echo $I18N_ADDON->msg('import'); ?></th>
```

Dabei wird die in der Datei install.inc.php definierte Sprachinstanz (\$I18N\_ADDON) genutzt und ausgegeben.

### Seiten (/pages)

Um im Layout der REDAXO Seiten nun die Seiten des Addons zu programmieren stehen Ihnen für die Navigation, sowie die Abschlussleiste folgender Syntax zur Verfügung

Erstellen der Navigation und grauen Titelleiste einer Seite:

```
<?
include $REX[INCLUDE_PATH]."/layout_redaxo/top.php";
title($I18N_MESSAGES->msg("title"), "");
?>
```

Abschlussleiste:

```
<?
include $REX[INCLUDE_PATH]."/layout_redaxo/bottom.php";
?>
```

## C5. Specials

Unter dieser Rubrik stehen Informationen zur Verfügung, die sich auf die Einstellungen in der Datei master.inc.php beziehen. Sie wurde in der Version 3.0 gegenüber den Versionen 2.7\* umgestaltet und erweitert.

| Spezielle Features  |  |
|---|--|
| Einstellungen   Sprachen   Typen  |  |
| Features  |  |
| <p><b>Regeneriere Artikel &amp; Cache</b><br/>Kategorien, Artikel und Templates werden neu generiert. Der gesamte Cache wird gelöscht. Kann länger dauern. Bitte den Vorgang nicht abbrechen.</p> <p><b>Linkchecker</b><br/>Wenn sie überprüfen wollen ob alle Links [REX_LINK] funktionieren, dann klicken Sie bitte hier.</p> <p><b>Setup</b><br/>Hier können Sie das Setup erneut starten. Sollten Sie nur im Notfall machen und ist unter normalen Umständen nicht nötig.</p> | <p><b>Allgemeine Informationen:</b><br/>\$REX['version']:<br/>\$REX['SERVER']:<br/>\$REX['SERVERNAME']:</p> <p><b>Datenbank[1] kann nur über</b><br/>\$DB['1']['HOST']:<br/>\$DB['1']['LOGIN']:<br/>\$DB['1']['PSW']:<br/>\$DB['1']['NAME']:</p> <p><b>Sonstiges</b><br/>\$REX['WWW_PATH']:<br/>\$REX['INCLUDE_PATH']:<br/>\$REX['error_emailaddress']:<br/>\$REX['STARTARTIKEL_ID']:<br/>\$REX['LANG']:<br/>\$REX['MOD_REWRITE']:</p> |

### C5.01 Main Preferences (Version 3.0)

Ab Version 3.1 heißt dieser Punkt **"Einstellungen"**. Main Preferences/Einstellungen enthält drei Themen.

#### Regeneriere Artikel & Cache

Über die Funktionen werden Informationen zu Kategorien, Artikeln und Templates neu erstellt. Dazu werden auch die gecachten Dateien gelöscht.

Wie im Manual oder in den Demos beschrieben, werden Informationen zu Kategorien und Artikeln, z.B. in der Navigation einer Webseite nicht über SQL-Anfragen gelöst, sondern aus generierten Dateien gelesen. Die Informationen werden beim Arbeiten in REDAXO automatisch angelegt und für eine schnelle Ausgabe bereitgestellt.

Die Dateien werden in den Ordnern articles, categories und templates im Ordner "redaxo/include/generated/" erstellt.

REDAXO nutzt bei der Ausgabe von Webseiten diese Informationen für eine schnelle und datenbankunabhängige Darstellung.

## Linkchecker

Wenn sie überprüfen wollen ob alle Links [REX\_LINK] funktionieren, dann können Sie das hier automatisch testen lassen..

## Setup (ab Version 3.0)

Hier können Sie das Setup erneut starten.

## Einstellungen (rechte Spalte)

Auf der rechten Seite können Einstellungen zur Webseite editiert und eingesehen werden.

`$REX[STARTARTIKEL_ID]` verweist hier auf den Artikel, der bei der Eingabe der URL als erster Artikel angezeigt wird.

The screenshot shows the 'Allgemeine Informationen' (General Information) section of the REDAXO preferences page. It contains several configuration fields:

- `$REX[version]`: "3.0"
- `$REX[SERVER]`: redaxo.de
- `$REX[SERVERNAME]`: REDAXO-Demo
- Datenbank[1] kann nur über das Setup geändert werden.**
- `$DB[1][HOST]`: (empty)
- `$DB[1][LOGIN]`: (empty)
- `$DB[1][PSW]`: (empty)
- `$DB[1][NAME]`: (empty)
- Sonstiges**
- `$REX[WWW_PATH]`: ""
- `$REX[INCLUDE_PATH]`: ../redaxo/include
- `$REX[error_emailaddress]`: jan.kristinus@pergopa.de
- `$REX[STARTARTIKEL_ID]`: 1
- `$REX[LANG]`: de\_de
- `$REX[MOD_REWRITE]`: TRUE
- Ändern

main\_preferences.gif

## C5.02 Sprachen (ab Version 3.0)

Redaxo bietet einem die Möglichkeit, mehrere Sprachen für eine Präsentation bequem zu verwalten. In diesem Unterpunkt können beliebig viele Sprachen angelegt werden.

Das System ist sehr flexibel. Die Sprachen können vor der Festlegung der Kategorien und Artikel angelegt werden. Es ist aber auch möglich, nach Festlegung der Seitenstruktur und der Inhalte noch Sprachen zu ergänzen.

Es wird immer die aktuelle Struktur für jedes Land gespiegelt. D.h. die Kategorien und Artikel sind in jedem Land identisch angelegt. Die Bezeichnungen der Kategorien und Artikel werden zunächst in allen Sprachen übernommen. Sie können dann entsprechend angepaßt werden. Wenn ein Artikel in einer Sprache neu angelegt wird, geschieht dies automatisch auch in allen anderen Sprachen.

Neu angelegte Kategorien und Artikel sind standardmäßig „offline“ gesetzt. Es ist möglich, diese Einstellung in jedem Land separat einzustellen. D. h. eine Seite, die in einer Sprache keine Relevanz hat, muß in dieser Sprache nicht mit Inhalten gefüllt und angezeigt werden.

### Neue Sprache anlegen

Zum Anlegen einer Sprache klickt man auf das „+“-Symbol, gibt den Namen der Sprache an und legt eine ID fest. Jede Sprache erhält eine eindeutige ID, über die die Navigation gesteuert wird. Die aktuelle Sprach-ID ist immer in REX['CUR\_CLANG'] gespeichert.

| +   | ID | Beschreibung/Name | -      |
|-----|----|-------------------|--------|
| add | 2  | dänisch           | submit |
|     | 0  | deutsch           |        |
|     | 1  | english           |        |

specials\_sprachen.gif

Der Wechsel zwischen den Sprachen erfolgt, indem man auf die gewünschte Sprach-ID verlinkt.

Beispiel:

```
echo "<a href=".rex_getUrl(20).">mein text</a>";
```

Link zum Artikel mit der id = 20; die Sprache wird beibehalten

```
echo "<a href=".rex_getUrl(20, 2).">mein text</a>";
```

Link zum Artikel mit der id = 20; die Sprache wird auf Sprach-id=2 gewechselt

```
echo "<a href=".rex_getUrl('', '', array('name' => 'peter', 'param2' => 'value2')).">mein text</a>";
```

Link zum aktuellen Artikel; die Sprache wird beibehalten; Der Link wird mit den Parametern *name* und *param2* erweitert.

Einzelne Sprachen können jederzeit gelöscht werden. Diese Funktion sollte man mit Vorsicht genießen, da hier „Tabula rasa“ gemacht wird. Mühselig eingepflegte Inhalte kann man dabei mit einem Knopfdruck ins Nirwana schicken.

## UTF-8-Codierung

Redaxo unterstützt in der Version 3.0 noch keine UTF-8-Codierung. Hier ist die Voraussetzung, dass sich diese Sprachen mit ISO-8858-1 bzw. -2 darstellen lassen. Ab der Version 3.1 ist UTF-8-Codierung möglich.

### Version 3.1:

In der Version 3.1 müssen folgende Einstellungen vorgenommen werden:

Unter `/redaxo/include/lang` findet man die Language-Dateien (z. B. `'de_de_lang'`). In diesen Dateien muß die charset-Einstellung folgendermaßen geändert werden:

```
htmlcharset = utf-8
```

### Version 3.2:

In der Version 3.2 wurden die Sprachdateien überarbeitet und ergänzt. Die oben angegebene Einstellung ist hier nicht mehr nötig. Bei dieser Version muß allerdings beachtet werden, dass Frontend und Backend dieselben Einstellungen haben müssen. ISO und utf-8 im Mix geht nicht. Die Festlegung für das Backend erfolgt unter "Specials". Sobald man im Template als charset utf-8 angibt, muß man auch im Backend eine utf-8-Spracheinstellung auswählen.



### Version 3.1 und 3.2

In den einzelnen Templates muß folgendes angegeben sein:

```
<?php
header('Content-Type: text/html; charset=utf-8');
?>
<!DOCTYPE HTML .....>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
</head>
```

**Wenn eine Seite UTF-8 konform sein soll, dann darf nur noch `htmlspecialchars()` verwendet werden! (`htmlentities()` ist Tabu!)**

### *Inhalte von Iso auf Utf-8 konvertieren*

Eine Beschreibung, wie man vorhandene Seiten, die in dem Zeichensatz iso-8859-1 erstellt worden sind, auf utf-8 konvertiert, findet man hier:

<http://www.redaxo.de/227-0-c5-02-01-iso-auf-utf-8-konvertieren.html>

## C5.03 Typen (ab Version 3.0)

Auf dieser Seite werden notwendige Artikeltypen erstellt und verwaltet. Mit Hilfe der Artikeltyps können Artikel gekennzeichnet und unterschieden werden. Dies kann z.B. hilfreich bei Suchanfrage sein oder zur Kennzeichnung von Seiten, die im geschlossenen Bereich liegen.

### Neuen Artikeltyp anlegen

Um einen neuen Artikeltyp anzulegen, klickt man auf das "+"-Zeichen und trägt in die Eingabefelder den Artikeltyp-Namen und eine Beschreibung ein. Die ID wird automatisch vergeben. Diese Eingaben werden in der Tabelle "rex\_article\_type" gespeichert.



The screenshot shows a web interface for managing article types. At the top, there is a header 'Spezielle Features' and a navigation menu with 'Einstellungen', 'Sprachen', and 'Typen'. Below the menu is a table with the following structure:

|   | ID | Artikeltyp                                 | Beschreibung                                       |  |
|---|----|--|--|--|
| + | 1  | Standard                                   | Zugriff für alle                                   |  |
|   | 2  | <input type="text" value="Artikeltyp II"/> | <input type="text" value="Zugriff nur für Admin"/> | <input type="button" value="Ändern"/> <input type="button" value="Löschen"/> |

### Zuweisung des Typs zu einem Artikel

Bei der Bearbeitung eines Artikels kann man unter der Rubrik "Metadaten/Sonstiges" die Zuweisung eines Artikels zu einem bestimmten Typ vornehmen (s. Kapitel C1.04 Metadaten/Sonstiges). Die Zuweisung wird in der Tabelle "rex\_article" in der Spalte "type\_id" gespeichert.

### Einsatz von Artikel-Typen

In den Modulen kann die Typ-ID verwendet werden, um bestimmte Eigenschaften zu programmieren. Das heißt, für die Arbeit mit Typen braucht man entweder Programmierkenntnisse – oder fertige Module freundlicher Mitmenschen.

## C6. Object Oriented Framework

Sie können auf die Informationen in den Tabellen, die REDAXO verwendet, über SQL-Befehle zugreifen oder wie auf den folgenden Seite beschrieben, über ein Objekt-Orientiertes Framework. In REDAXO 3.x greifen diese Funktionen auf die generierten Dateien zurück und verbessern so die Performance. (REDAXO 2.7.x greift dabei über SQL-Befehl auf die Inhalte der Tabelle zurück).

Praktische Beispiele zur Verwendung findet man in den Demos, zb. in der Abbildung von Navigationen. Oder hier in Bereich [C.C1. Template/Navigationen](#). Alle verwendete Funktionen finden sich in den folgenden Klassen.

Die Klasse und enthaltenen Funktionen zu diesem Konzept finden Sie in den folgenden Dateien:

- class.oocategory.inc
- class.ooarticle.inc
- class.ooarticleslice.inc
- class.oomedia.inc
- class.oomediacategory.inc
- 

Im Folgenden werden Funktionen und Beispiele gezeigt:

[OOCategory](#)

[OOArticle](#)

[OOArticleSlice](#)

[OOMedia](#)

[OOMediaCategory](#)

### OOF-Übersicht zum Download

Eine sehr gute Zusammenstellung aller Klassen mit ihren Funktionen und Variablen findet man unter folgender Adresse:

[www.webbude.com/89-0-oof-uebersicht.html](http://www.webbude.com/89-0-oof-uebersicht.html)

## C6.01 OOCategory

Die Beispiele wurden mit der Community Demo erstellt. Einige Befehle (#) sind noch nicht beschrieben.

Übersicht:

- [OO Category-Klasse](#)
- [getCategoryById\(\)](#)
- [searchCategoriesByName\(\)](#)
- [getRootCategories\(\)](#)
- [getChildren\(\)](#)
- [getParent\(\)](#)
- [# isParent\(\)](#)
- [getArticles\(\)](#)
- [# getArticlesByDate\(\)](#)
- [getStartArticle\(\)](#)
- [getId\(\)](#)
- [getName\(\)](#)
- [getPriority\(\)](#)
- [getUrl\(\)](#)
- [isOnline\(\)](#)
- [toString\(\)](#)

### **OOCategory-Klasse**

#### **Verfügbarkeit**

ab 2.7

#### **Beschreibung**

Mit der Kategorie-Klasse können Sie Informationen zu den einzelnen Kategorie erhalten, wie ID, Name, Description, Func, Prior, Path, Status.

#### ***getCategoryById (2)***

#### **Verfügbarkeit**

ab 2.7

#### **Verwendung**

```
$cat = OOCategory::getCategoryById('id');
```

#### **Parameter**

id, Kategorie-ID

#### **Rückgaben**

OOCategory (\_id, \_name, \_description, \_func, \_re\_category\_id, \_prior, \_path, \_status)

#### **Beschreibung**

Methode; gibt Informationen zu der über den Parameter id angegebenen Kategorie zurück.

#### **Beispiel**

```
<?
$cat = OOCategory::getCategoryById(2);
print_r($cat);
?>
```

Ausgabe:

```
oocategory Object
(
  [_id] => 2
  [_name] => BOARDS
  [_description] =>
  [_func] =>
  [_re_category_id] => 0
  [_prior] => 7
  [_path] =>
  [_status] => 1
)
```

Auf die einzelnen Attribute können Sie z.B. mit weiteren Methoden zugreifen:

```
<?
echo $cat -> getName();
?>
```

Ausgabe:\n  
BOARDS

### ***searchCategoriesByName()***

#### **Verfügbarkeit**

ab 2.7

#### **Verwendung**

```
$cat = OOCategory::searchCategoriesByName(name);
$cat = OOCategory::searchCategoriesByName("%Name%");
```

Sucht nach Namen die offline gesetzt sind.  
OOCategory::searchCategoriesByName("%Name%", 1);

#### **Parameter**

name, Kategorie-Name; optional \$ignore\_offlines;

#### **Rückgaben**

Array mit ein oder mehreren, gefunden OOCategory?.

#### **Beschreibung**

Methode; gibt die Suchergebnisse nach prior gelistet zurück.

#### **Beispiel**

In diesen Beispiel wird nach der Kategorie Redaxo gesucht und eine Kategorie gefunden. Sie können dabei den vollen Namen oder wie in einee SQL-Anfrage "LIKE '%...&'" die Kategorie suchen.

```
$cat = OOCategory::searchCategoriesByName("%Mein%");
print_r($cat);
```

Ausgabe:

```
Array
(
  [0] => oocategory Object
  (
    [_id] => 13
```

```
[_name] => ALLGEMEIN
[_description] =>
[_func] =>
[_re_category_id] => 2
[_prior] => 2
[_path] => -2
[_status] => 1
)

[1] => oocategory Object
(
  [_id] => 6
  [_name] => MEIN GÄSTEBUCH
  [_description] =>
  [_func] =>
  [_re_category_id] => 0
  [_prior] => 12
  [_path] =>
  [_status] => 1
)

[2] => oocategory Object
(
  [_id] => 4
  [_name] => MEIN PROFIL
  [_description] =>
  [_func] =>
  [_re_category_id] => 0
  [_prior] => 11
  [_path] =>
  [_status] => 1
)

[3] => oocategory Object
(
  [_id] => 7
  [_name] => MEINE NACHRICHTEN
  [_description] =>
  [_func] =>
  [_re_category_id] => 0
  [_prior] => 13
  [_path] =>
  [_status] => 1
)

)
```

### ***getRootCategories()***

**Verfügbarkeit**  
ab 2.7

### **Verwendung**

```
$cat = OOCategory:: getRootCategories();
```

Rückgabe aller Root-Kategorien online  

```
$cat = OOCategory:: getRootCategories(1);
```

#### Parameter

optional, 1/0 | online/offline

#### Rückgaben

Array mit alle Root-Kategorien

#### Beschreibung

Methode; gibt alle Kategorien im Root nach prior sortiert zurück. Ist der Parameter auf true gesetzt, werden alle Kategorien die Status online haben, ausgegeben.

#### Beispiel

```
<?  
$cat = OOCategory:: getRootCategories();  
print_r($cat);  
?>
```

#### Ausgabe:

```
Array  
(  
[0] => oocategory Object  
(  
[_id] => 1  
[_name] => HOME  
[_description] =>  
[_func] =>  
[_re_category_id] => 0  
[_prior] => 5  
[_path] =>  
[_status] => 1  
)  
  
[1] => oocategory Object  
(  
[_id] => 3  
[_name] => NEWS  
[_description] =>  
[_func] =>  
[_re_category_id] => 0  
[_prior] => 6  
[_path] =>  
[_status] => 1  
)  
  
...  
  
[11] => oocategory Object(  
[_id] => 11  
[_name] => IMPRESSUM  
[_description] =>  
[_func] =>  
[_re_category_id] => 0  
[_prior] => 16  
[_path] =>
```

```
[_status] => 1
))
```

## **getChildren()**

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$cat->getChildren();
```

### **Parameter**

optional \$ignore\_offlines;

### **Rückgaben**

Array mit oocategory Objects

### **Beschreibung**

Methode; gibt ein Array mit Subkategorien zurück, sortiert nach prior

### **Beispiel**

```
$cat = OOCategory::getCategoryById(2);
$cat2 = $cat->getChildren();
print_r($cat2);
```

Ausgabe:

```
Array(
  [0] => oocategory Object(
    [_id] => 13
    [_name] => ALLGEMEIN
    [_description] =>
    [_func] =>
    [_re_category_id] => 2
    [_prior] => 2
    [_path] => -2
    [_status] => 1
  )

  [1] => oocategory Object(
    [_id] => 14
    [_name] => LERN MICH KENNEN
    [_description] =>
    [_func] =>
    [_re_category_id] => 2
    [_prior] => 3
    [_path] => -2
    [_status] => 1
  )
)
```

## **getParent()**

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
oocategory->getParent();
```

### **Parameter**

keine

**Rückgaben**

oocategory Object

**Beschreibung**

Methode; gibt Parent Kategorie zurück

**Beispiel**

```
<?  
$cat = OOCategory::getCategoryById(13);
```

```
$cat2 = $cat->getParent();  
print_r($cat2);
```

```
?>
```

Ausgabe:

```
oocategory Object  
(  
  [_id] => 2  
  [_name] => BOARDS  
  [_description] =>  
  [_func] =>  
  [_re_category_id] => 0  
  [_prior] => 7  
  [_path] =>  
  [_status] => 1  
)
```

## **getArticles()**

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
oocategory->getArticles()
```

### **Parameter**

&#x20;\$ignore\_offlines: offline/online, 0/1

### **Rückgaben**

Array mit Artikeln einer Kategorie

### **Beschreibung**

Methode; gibt Array mit Artikel einer Kategorie zurück, mit Parameter \$ignore\_offlines auf 1, werden alle Artikel mit Status 0 aus dem Array entfernt

### **Beispiel**

```
$cat = OOCategory::getCategoryById(3);  
$newArt = $cat->getArticles();  
print_r($newArt);
```

### **Ausgabe:**

Array

```
(  
[0] => oarticle Object  
(  
  [_id] => 3  
  [_name] => NEWS  
  [_beschreibung] =>  
  [_attribute] =>  
  [_file] =>  
  [_category_id] => 3  
  [_type_id] => 1  
  [_startpage] => 1  
  [_prior] => 1  
  [_path] => -3  
  [_status] => 1  
  [_online_von] => 20040713  
  [_online_bis] => 20100101  
  [_erstelldatum] => 20040713  
  [_suchbegriffe] =>  
  [_template_id] => 1  
  [_checkbox01] => 0  
  [_checkbox02] => 0  
  [_checkbox03] => 0  
  [_checkbox04] => 0  
)  
[1] => oarticle Object  
(  
  [_id] => 20  
  [_name] => Aktuelle Infos 1  
  [_beschreibung] =>  
  [_attribute] =>  
  [_file] =>  
  [_category_id] => 3  
  [_type_id] => 1  
  [_startpage] => 0
```

```
[_prior] => 2
[_path] => -3
[_status] => 1
[_online_von] => 20040715
[_online_bis] => 20100101
[_erstelldatum] => 20040715
[_suchbegriffe] =>
[_template_id] => 1
[_checkbox01] => 0
[_checkbox02] => 0
[_checkbox03] => 0
[_checkbox04] => 0
)
)
```

### ***getStartArticle()***

#### **Verfügbarkeit**

ab 2.7

#### **Verwendung**

```
oocategory->getStartArticle()
```

#### **Parameter**

keine

#### **Rückgaben**

gibt ooarticle objekt zurück

#### **Beschreibung**

Methode; gibt das Artikel-Objekt zurück, mit den Eigenschaften der Artikels (id, name, beschreibung, attribute, file, category\_id, type\_id, startpage, prior, path, online\_von, online\_bis, erstelldatum, suchbegriffe, template\_id, checkbox01-04

#### **Beispiel**

```
<?
$cat = OOCategory::getCategoryById(1);
print_r($cat->getStartArticle());
?>
```

Ausgabe:

```
ooarticle Object
(
  [_id] => 1
  [_name] => Home
  [_beschreibung] =>
  [_attribute] =>
  [_file] =>
  [_category_id] => 1
  [_type_id] => 1
  [_startpage] => 1
  [_prior] => 2
  [_path] => -1
```

```
[_status] => 1
[_online_von] => 20040713
[_online_bis] => 20100101
[_erstelldatum] => 20040713
[_suchbegriffe] =>
[_template_id] => 1
[_checkbox01] => 0
[_checkbox02] => 0
[_checkbox03] => 0
[_checkbox04] => 0
)
```

### ***getId()***

#### **Verfügbarkeit**

ab 2.7

#### **Verwendung**

```
oocategory->getId()
```

#### **Parameter**

keine

#### **Rückgaben**

ID der Kategorie

#### **Beschreibung**

Methode; gibt die ID der Kategorie zurück

#### **Beispiel**

```
<?
$cat = OOCategory::getCategoryById(1);
echo $cat->getId();
?>
```

Ausgabe:

1

***getName ()*****Verfügbarkeit**

ab 2.7

**Verwendung**

```
oocategory->getName ()
```

**Parameter**

keine

**Rückgaben**

Name der Kategorie, string

**Beschreibung**

Methode; gibt den Namen der Kategorie zurück.

**Beispiel**

```
<?
$cat = OOCategory::getCategoryById(1);
echo $cat->getName ();
?>
```

Ausgabe:

HOME

***getPriority ()*****Verfügbarkeit**

ab 2.7

**Verwendung**

```
oocategory->getPriority ()
```

**Parameter**

keine

**Rückgaben**

Prior

**Beschreibung**

Methode; gibt die Priorität einer Kategorie zurück, kann sich von der angezeigten Prio in REDAXO unterscheiden.

**Beispiel**

```
<?
$cat = OOCategory::getCategoryById(1);
print_r($cat);
echo $cat->getPriority ();
?>
```

Ausgabe:

5

***getUrl()*****Verfügbarkeit**

ab 2.7

**Verwendung**

```
oocategory->getUrl()
```

**Parameter**

keine

**Rückgaben**

Url Startartikel Kategorie

**Beschreibung**

Methode; gibt die Url zu dem Startartikel einer Kategorie an z.B. index.php?article\_id=25

**Beispiel**

```
<?
$cat = OOCategory::getCategoryById(3);
echo $cat->getUrl();
?>
```

Ausgabe:

index.php?article\_id=3

***isOnline()*****Verfügbarkeit**

ab 2.7

**Verwendung**

```
oocategory->isOnline()
```

**Parameter**

keine

**Rückgaben**

true (1) – online, false (0) – offline

**Beschreibung**

Methode; gibt den Status der Kategorie zurück, Rückgabewert 1 oder 0

**Beispiel**

```
<?
$cat = OOCategory::getCategoryById(1);
echo $cat->isOnline();
?>
```

Ausgabe:

1

***toString()*****Verfügbarkeit**

ab 2.7

**Verwendung**

```
oocategory->toString();
```

**Parameter**

keine

**Rückgaben**

String, Information Kategorie (Debugging)

**Beschreibung\***

Methode; gibt einen String mit Information zu einem Artikel zurück. (Kategorie-Id, Kategorie-Name, Status)

**Beispiel**

```
<?  
$cat = OOCategory::getCategoryById(3);  
echo $cat->toString();  
?>
```

Ausgabe:

```
Category: 3, NEWS, online
```

## C6.02 OOArticle

Die Beispiele wurden mit der Community Demo erstellt.

Übersicht:

- [OOArticle - Klasse](#)
- [getArticleById\(\)](#)
- [searchArticlesByName\(\)](#)
- [getArticlesByType\(\)](#)
- [getCategory\(\)](#)
- [getSlicesOfType\(\)](#)
- [getId\(\)](#)
- [getName\(\)](#)
- [getDescription\(\)](#)
- [getUrl\(\)](#)
- [isOnline\(\)](#)
- [toString\(\)](#)

### OOArticle-Klasse

#### Verfügbarkeit

ab 2.7

#### Beschreibung

Klasse; Mit der Article-Klasse können Sie Informationen zu den einzelnen Artikeln erhalten, wie ID, Name, Beschreibung, Attribute, Typ, Prior, Path, Status, Startpage, Kategorie ...

### *getArticleById()*

#### Verfügbarkeit

ab 2.7

#### Verwendung

```
$cat = OOArticle::getArticleById('id');
```

#### Parameter

id, Artikel-ID

#### Rückgaben

oarticle Object, (id, name, beschreibung, attribute, file, category\_id, type\_id, startpage, prior, path, status, online\_von, online\_bis, erstelldatum, suchbegriffe, template\_id, checkbox01, checkbox02, checkbox03, checkbox04)

#### Beschreibung

Methode; gibt Informationen zu dem, über den Parameter id angegebenen, Artikel zurück.

#### Beispiel

```
<?
$art = OOArticle::getArticleById(21);
print_r($art);
?>
```

Ausgabe:

```
ooarticle Object
(
  [_id] => 21
  [_name] => test
  [_beschreibung] =>
  [_attribute] =>
  [_file] =>
  [_category_id] => 0
  [_type_id] => 1
  [_startpage] => 0
  [_prior] => 3
  [_path] =>
  [_status] => 0
  [_online_von] => 20050629
  [_online_bis] => 20100101
  [_erstelldatum] => 20050629
  [_suchbegriffe] =>
  [_template_id] => 1
  [_checkbox01] => 0
  [_checkbox02] => 0
  [_checkbox03] => 0
  [_checkbox04] => 0
)
```

Auf die einzelnen Attribute können Sie z.B. mit weiteren Methoden zugreifen:

```
%block bgcolor=#eee border='1px solid black' padding=6px% [=
<?
echo $art -> getName();
?>
```

Ausgabe:

MEIN PROFIL

***searchArticlesByName()*****Verfügbarkeit**

ab 2.7

**Verwendung**

```
OOArticle::searchArticlesByName("%Name%");
```

**Parameter**

name, Artikel-Name;

**Rückgaben**

Array, gefundene Artikel der definierten Suche

**Beschreibung**

Methode; gibt die Suchergebnisse in einem Array gelistet zurück.

**Beispiel**

In diesen Beispiel wird nach einem Artikel mit dem Begriffe "Mein" gesucht. Sie können dabei den vollen Namen oder wie in eine SQL-Anfrage "LIKE '%...&'" suchen.

```
<?
$article_search = OOArticle::searchArticlesByName ("%Mein%");
print_r($article_search);
```

```
?>
```

Ausgabe:

```
Array
(
  [0] => ooarticle Object
  (
    [_id] => 4
    [_name] => MEIN PROFIL
    [_beschreibung] =>
    [_attribute] =>
    [_file] =>
    [_category_id] => 4
    [_type_id] => 3
    [_startpage] => 1
    [_prior] => 1
    [_path] => -4
    [_status] => 1
    [_online_von] => 20040713
    [_online_bis] => 20100101
    [_erstelldatum] => 20040713
    [_suchbegriffe] =>
    [_template_id] => 1
    [_checkbox01] => 0
    [_checkbox02] => 0
    [_checkbox03] => 0
    [_checkbox04] => 0
  )
  [1] => ooarticle Object
  (
    [_id] => 6
    [_name] => MEIN GÄSTEBUCH
    [_beschreibung] =>
    [_attribute] =>
```

```
[_file] =>
[_category_id] => 6
[_type_id] => 3
[_startpage] => 1
[_prior] => 1
[_path] => -6
[_status] => 1
[_online_von] => 20040713
[_online_bis] => 20100101
[_erstelldatum] => 20040713
[_suchbegriffe] =>
[_template_id] => 1
[_checkbox01] => 0
[_checkbox02] => 0
[_checkbox03] => 0
[_checkbox04] => 0
)

....
)
```

### ***getArticlesByType(id)***

#### **Verfügbarkeit**

ab 2.7

#### **Verwendung**

```
OOArticle::getArticlesByType(id)
```

#### **Parameter**

Typ, Artikel-Typ (Specials)

#### **Rückgaben**

Array mit oarticle Objects der gesuchten Artikel

#### **Beschreibung**

Methode; gibt ein Array mit oarticle Objects zurück, die den gesuchten Typ besitzen.

#### **Beispiel**

Suche nach Artikeln mit dem Typ 3 (User).

```
<?
$article_search = OOArticle::getArticlesByType(3);
print_r($article_search);
?>
```

Ausgabe:

```
Array
(
  [0] => ooarticle Object
  (
    [_id] => 4
    [_name] => MEIN PROFIL
    [_beschreibung] =>
    [_attribute] =>
    [_file] =>
    [_category_id] => 4
    [_type_id] => 3
    [_startpage] => 1
    [_prior] => 1
    [_path] => -4
    [_status] => 1
    [_online_von] => 20040713
    [_online_bis] => 20100101
    [_erstelldatum] => 20040713
    [_suchbegriffe] =>
    [_template_id] => 1
    [_checkbox01] => 0
    [_checkbox02] => 0
    [_checkbox03] => 0
    [_checkbox04] => 0
  )

  [1] => ooarticle Object
  (
    [_id] => 6
    [_name] => MEIN GÄSTEBUCH
    [_beschreibung] =>
    [_attribute] =>
    [_file] =>
    [_category_id] => 6
    [_type_id] => 3
    [_startpage] => 1
    [_prior] => 1
    [_path] => -6
    [_status] => 1
    [_online_von] => 20040713
    [_online_bis] => 20100101
    [_erstelldatum] => 20040713
    [_suchbegriffe] =>
    [_template_id] => 1
    [_checkbox01] => 0
    [_checkbox02] => 0
    [_checkbox03] => 0
    [_checkbox04] => 0
  )

  ...
)
```

## ***getCategory()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
ooarticle->getCategory()
```

### **Parameter**

keine

### **Rückgaben**

oocategory Object ( *\_id*, *\_name*, *\_description*, *\_func*, *\_re\_category\_id*, *\_prior*, *\_path*, *\_status*)

### **Beschreibung**

Methode; gibt oocategory Objekt zurück mit den jeweiligen Eigenschaften. Weitere Informationen zu den Funktionen eines oocategory Objekts finden sie unter C6.01 OOCategory .

### **Beispiel**

```
<?
$article = OOArticle::getArticleById(20);
$cat = $article->getCategory();
print_r($cat);
?>
```

Ausgabe:

```
oocategory Object
(
  [_id] => 3
  [_name] => NEWS
  [_description] =>
  [_func] =>
  [_re_category_id] => 0
  [_prior] => 6
  [_path] =>
  [_status] => 1
)
```

## ***getSlicesOfType()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
ooArticle->getSlicesOfType($type_id)
```

### **Parameter**

*type\_id*

### **Rückgaben**

Array ooArticleSlice

### **Beschreibung**

Methode; gibt ein Array von ooArticleSlice-Objekten, in der Reihenfolge der Verwendung, eines Artikel zurück.

## ***getId()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
oarticle->getId();
```

**\*Parameter\***

**\*Rückgaben\***

Artikel-Id

**\*Beschreibung\***

Methode; gibt die Artikel-Id des oarticle Objekts zurück

**\*Beispiel\***

```
<?
$article_search = OoArticle::searchArticlesByName ("%Mein%");

foreach($article_search as $article){
echo "<a href=index.php?article_id=" . $article->getId() .">" . $article-
>getName() . "</a><br>" ;
}
?>
```

Ausgabe:

```
<a href=index.php?article_id=4>MEIN PROFIL</a><br>
<a href=index.php?article_id=6>MEIN GÄSTEBUCH</a><br>
<a href=index.php?article_id=7>MEINE NACHRICHTEN</a><br>
<a href=index.php?article_id=17>ALLGEMEIN</a><br>
```

## ***getName()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
oarticle->getName();
```

**Parameter**

keine

**Rückgaben**

Artikelname

**Beschreibung**

Methode; gibt den Artikelnamen eines oarticle Objekts zurück.

**Beispiel**

Suche nach Artikel mit dem Namen "Mein", danach Ausgabe der Artikelnamen der gefundenen Artikel.

```
<?
$article_search = OOArticle::searchArticlesByName ("%Mein%");

for ($i=0; $i<count($article_search); $i++)
{
$name = $article_search[$i]->getName() ."<br>";
echo $name ;
}
?>
```

Ausgabe:

```
MEIN PROFIL
MEIN GÄSTEBUCH
MEINE NACHRICHTEN
ALLGEMEIN
```

***getDescription()*****Verfügbarkeit**

ab 2.7

**Verwendung**

```
ooarticle->getDescription();
```

**Parameter**

keine

**Rückgaben**

Beschreibung des Artikels, Metadaten

**Beschreibung**

Methode; gibt den Wert der Beschreibung (Metadaten) zurück.

**Beispiel**

```
<?
$article = OOArticle::getArticleById(1);
echo $article->getDescription();

?>
```

Ausgabe:

```
Homepage Demo Community
```

## ***getUrl()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
ooarticle->getUrl()
```

### **Parameter**

#### **Rückgaben**

Url, index.php?article\_id=

#### **Beschreibung**

Methode; gibt die url der Artikels zurück. Die Rückgabe unterstützt auch mod\_rewrite. Parameter als assoziativer Array übergeben, werden an die Url angefügt.

```
$param = array("order" => "123", "name" => "horst");  
$article->getUrl($param);
```

```
index.php?article_id=1&order=123&name=horst
```

bei aktivierten mod\_rewrite:

```
/1-The_Article_Name?order=123&name=horst
```

### **Beispiel**

```
<ul>  
<?  
$article_search = OOArticle::searchArticlesByName ("%Mein%");  
  
foreach ($article_search as $article) {  
echo "<li><a href=\"" . $article->getUrl() . ">" . $article->getName() .  
"</a></li>";  
}  
  
?>  
</ul>
```

Ausgabe:

```
<ul>  
<li><a href=index.php?article_id=4>MEIN PROFIL</a></li>  
<li><a href=index.php?article_id=6>MEIN GÄSTEBUCH</a></li>  
<li><a href=index.php?article_id=7>MEINE NACHRICHTEN</a></li>  
<li><a href=index.php?article_id=17>ALLGEMEIN</a></li>  
</ul>
```

## *isOnline()*

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
ooarticle->isOnline();
```

### **Parameter**

keine

### **Rückgaben**

true, 1 wenn online

### **Beschreibung**

Methode; gibt Wert true/1 zurück, wenn Artikel den Status online hat.

### **Beispiel**

```
<?
$article = OOArticle::getArticleById(20);
echo $article->isOnline();
?>
```

Ausgabe:

```
1
```

## *toString()*

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
ooArticle->toString()
```

### **Parameter**

keine

### **Rückgaben**

String, Artikel Informationen (Debugging)

### **Beschreibung**

Methode; gibt einen String mit Information zu einem Artikel zurück. (Artikel-Id, Artikel-Name, Status)

### **Beispiel**

```
<?
$article = OOArticle::getArticleById(20);
$debugString = $article->toString();
```

```
echo $debugString;
```

```
?>
```

Ausgabe:

```
Article: 20, Aktuelle Infos 1, offline
```

## C6.03 OOArticleSlice

Die Beispiele wurden mit der Community Demo erstellt.

Übersicht:

- [OOArticleSlice-Klasse](#)
- [getArticleSliceById\(\)](#)
- [getFirstSliceForArticle\(\)](#)
- [getSlicesForArticleOfType\(\)](#)
- [getNextSlice\(\)](#)
- [getPreviousSlice\(\)](#)
- [fullTextSearch\(\)](#)
- [getArticle\(\)](#)
- [getId\(\)](#)
- [getValue\(\)](#)
- [getLink\(\)](#)
- [getLinkUrl\(\)](#)
- [getFile\(\)](#)
- [getFileUrl\(\)](#)
- [getHtml\(\)](#)
- [getPhp\(\)](#)

### **OOArticleSlice-Klasse**

Verfügbarkeit  
ab 2.7

Beschreibung

Klasse; Mit der ArticleSlice-Klasse können Sie Informationen zu den Variablen der einzelnen Blöcke, die zu einem Artikel gehören, erhalten. Diese Variablen sind id, re\_article\_slice\_id, value1 – value10, link1 – link10, file1 – file10, php, html, article\_id, modultypid.

Ein „Slice“ entspricht einem Block, der zu einem Artikel gehört.

#### **Hinweis:**

Die OOArticleSlice Klasse läuft direkt über die Datenbank läuft und ist daher im Moment noch nicht so performant ist. Die Anderen OO Klassen laufen über gecachte Dateien und haben daher eine ordentlich Performance.

## **getArticleSliceById()**

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$slice = OOArticleSlice::getArticleSliceById('id');
```

### **Parameter**

id, Slice-ID

### **Rückgaben**

OOArticleSlice-Objekt (id, re\_article\_slice\_id, value1 – value10, link1 – link10, file1 – file10, php, html, article\_id, modultypid)

### **Beschreibung**

Methode; gibt Informationen über den Article-Slice mit einer bestimmten ID zurück. Jeder Slice hat eine eindeutige ID.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getArticleSliceById(31);
print_r($slice);
?>
```

### **Ausgabe**

```
ooarticleslice Object (
  [_id] => 31
  [_re_article_slice_id] => 22
  [_value] => Array ( [1] => Ersparen Sie sich und ....
  [2] =>
  ...
  [10] =>
  )
  [_file] => Array ( [1] =>
  [2] =>
  [3] =>
  ...
  [10] =>
  )
  [_link] => Array ( [1] => 0
  [2] => 0
  [3] => 0
  ...
  [10] => 0 )
  [_php] =>
  [_html] =>
  [_article_id] => 1
  [_modultyp_id] => 2
)
```

Auf die einzelnen Attribute kann mit weiteren Methoden zugegriffen werden:

```
echo $slice -> getValue(1);
```

Ausgabe: Ersparen Sie sich und ....

## ***getFirstSliceForArticle()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$slice = OOArticleSlice::getFirstSliceForArticle('id');
```

### **Parameter**

id, Article-ID

### **Rückgaben**

OOArticleSlice-Objekt (id, re\_article\_slice\_id, value1 – value10, link1 – link10, file1 – file10, php, html, article\_id, modultypid)

### **Beschreibung**

Methode; gibt Informationen über den ersten Article-Slice eines vorgegebenen Artikels aus. Der erste Slice eines Artikels hat die re\_article\_slice\_id 0.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getFirstSliceForArticle(1);
print_r($slice);
?>
```

### **Ausgabe**

```
ooarticleslice Object (
  [_id] => 22
  [_re_article_slice_id] => 0
  [_value] => Array ( [1] => HOME
  [2] =>
  ...
  [10] =>
  )
  [_file] => Array ( [1] =>
  [2] =>
  ...
  [10] =>
  )
  [_link] => Array ( [1] => 0
  [2] => 0
  ...
  [10] => 0
  )
  [_php] =>
  [_html] =>
  [_article_id] => 1
  [_modultyp_id] => 6 )
```

## ***getSlicesForArticleOfType()***

### **Verfügbarkeit**

ab 2.7

## Verwendung

```
$slice = OOArticleSlice::getSlicesForArticleOfType(article_id, type_id);
```

## Parameter

article\_id = Article-ID, type\_id = Modultyp\_ID

## Rückgaben

OOArticleSlice-Objekt (id, re\_article\_slice\_id, value1 – value10, link1 – link10, file1 – file10, php, html, article\_id, modultypid)

## Beschreibung

Methode; gibt Informationen aus über Article-Slices, die zu einem bestimmten Artikel gehören und die einem vorgegebenen Modultyp entsprechen. Hier können auch mehrere Slices selektiert werden, wenn bei einem Artikel der gesuchte Modultyp mehrfach verwendet wurde.

## Beispiel

```
<?
$slice = OOArticleSlice::getSlicesForArticleOfType(1, 6);
print_r($slice);
?>
Ausgabe
Array ( [0] =>
ooarticleslice Object (
  [_id] => 22
  [_re_article_slice_id] => 0
  [_value] => Array ( [1] => HOME
[2] =>
[3] =>
...
[10] =>
)
  [_file] => Array ( [1] =>
[2] =>
[3] =>
...
[10] =>
)
  [_link] => Array ( [1] => 0
[2] => 0
[3] => 0
...
[10] => 0
)
  [_php] =>
  [_html] =>
  [_article_id] => 1
  [_modultyp_id] => 6
)
)
```

## *getNextSlice()*

## Verfügbarkeit

ab 2.7

## Verwendung

```
$slice = OOArticleSlice::getNextSlice();
```

## Parameter

keine

## Rückgaben

OOArticleSlice-Objekt (id, re\_article\_slice\_id, value1 – value10, link1 – link10, file1 – file10, php, html, article\_id, modultypid)

## Beschreibung

Methode; gibt Informationen aus über den Article-Slice, der auf einen bereits selektierten Slice folgt.

## Beispiel

```
<?
$slice1 = OOArticleSlice::getFirstSliceForArticle(27);
$slice2 = $slice1->getNextSlice();
$slice3 = $slice2->getNextSlice();
print_r($slice2);
?>
Ausgabe
oarticleslice Object (
  [_id] => 55
  [_re_article_slice_id] => 54
  [_value] => Array ( [1] => Streng dem definierten Wesen des Blindtextes folgend,
  [2] =>
  [3] =>
  ...
  [10] =>
  )
  [_file] => Array ( [1] =>
  [2] =>
  [3] =>
  ...
  [10] =>
  )
  [_link] => Array ( [1] => 0
  [2] => 0
  [3] => 0
  ...
  [10] => 0
  )
  [_php] =>
  [_html] =>
  [_article_id] => 27
  [_modultyp_id] => 2
  )
```

## ***getPreviousSlice()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$slice = OOArticleSlice::getPreviousSlice();
```

### **Parameter**

keine

### **Rückgaben**

OOArticleSlice-Objekt (id, re\_article\_slice\_id, value1 – value10, link1 – link10, file1 – file10, php, html, article\_id, modultypid)

### **Beschreibung**

Methode; gibt Informationen aus über den Article-Slice, der einem bereits selektierten Slice vorhergeht.

### **Beispiel**

```
$slice = OOArticleSlice::getArticleSliceById(31);  
$slice2 = $slice->getPreviousSlice();  
print_r($slice2);
```

#### Ausgabe

```
ooarticleslice Object ( [_id] => 22  
  [_re_article_slice_id] => 0  
  [_value] => Array ( [1] => HOME  
  [2] =>  
  [3] =>  
  ...  
  [10] => )  
  [_file] => Array ( [1] =>  
  [2] =>  
  [3] =>  
  ...  
  [10] => )  
  [_link] => Array ( [1] => 0  
  [2] => 0  
  [3] => 0  
  ...  
  [10] => 0  
  )  
  [_php] =>  
  [_html] =>  
  [_article_id] => 1  
  [_modultyp_id] => 6  
  )
```

## **getArticle()**

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$article = OOArticleSlice::getArticle();
```

### **Parameter**

keine

### **Rückgaben**

OOArticle-Objekt (id, name, beschreibung, attribute, file, category\_id, type\_id, startpage,prior, path, status, online\_von, online\_bis, erstelldatum, suchbegriffe, template\_id, checkbox 0 – 4)

### **Beschreibung**

Methode; gibt Informationen aus über den Artikel, zu dem ein selektierter Slice gehört.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getArticleSliceById(31);
$article = $slice->getArticle();
print_r($article);
?>
```

### **Ausgabe**

```
ooarticle Object ( [_id] => 1
[_name] => Home
[_beschreibung] =>
[_attribute] =>
[_file] =>
[_category_id] => 1
[_type_id] => 1
[_startpage] => 1
[_prior] => 2
[_path] => -1
[_status] => 1
[_online_von] => 20040713
[_online_bis] => 20100101
[_erstelldatum] => 20040713
[_suchbegriffe] =>
[_template_id] => 1
[_checkbox01] => 0
[_checkbox02] => 0
[_checkbox03] => 0
[_checkbox04] => 0
)
```

## ***getId()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$id = OOArticleSlice::getId();
```

### **Parameter**

keine

### **Rückgaben**

Slice-ID

### **Beschreibung**

Methode; gibt die ID des selektierten Slices aus.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getFirstSliceForArticle(1);
$id = $slice->getId();
print_r($id);
?>
```

Ausgabe

22

## ***getValue***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$value = OOArticleSlice::getValue(index);
```

### **Parameter**

Index-Wert (1 – 10)

### **Rückgaben**

String

### **Beschreibung**

Methode; gibt einen bestimmten Value eines selektierten Slices aus.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getFirstSliceForArticle(8);
$value = $slice->getValue(1);
print_r($value);
?>
```

Ausgabe

OPEN BOARD

## ***getLink()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$link = OOArticleSlice::getLink(index);
```

### **Parameter**

Index-Wert (1 – 10)

### **Rückgaben**

Article\_ID eines internen Links

### **Beschreibung**

Methode; gibt die Artikel-ID eines indizierten internen Links des selektierten Sclices aus.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getFirstSliceForArticle(8);
$link = $slice->getLink(1);
print_r($link);
?>
```

Ausgabe

12

## ***getLinkUrl()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$url = OOArticleSlice::getLinkUrl(index);
```

### **Parameter**

Index-Wert (1 – 10)

### **Rückgaben**

URL eines interner Links

### **Beschreibung**

Methode; gibt die URL eines indizierten internen Links des selektierten Sclices aus.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getArticleSliceById(52);
$url = $slice->getLinkUrl(1);
print_r($url);
?>
```

Ausgabe

index.php?article\_id=2

## ***getFile()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$file = OOArticleSlice::getFile(index);
```

### **Parameter**

Index-Wert (1 – 10)

### **Rückgaben**

Dateiname

### **Beschreibung**

Methode; gibt den Dateinamen eines indizierten Files des selektierten Slices aus.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getArticleSliceById(52);
$file = $slice->getFile(1);
print_r($file);
?>
```

Ausgabe  
banner.jpg

## ***getFileUrl()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$file = OOArticleSlice::getFileUrl(index);
```

### **Parameter**

Index-Wert (1 – 10)

### **Rückgaben**

URL

### **Beschreibung**

Methode; gibt die URL eines indizierten Files des selektierten Slices aus.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getArticleSliceById(52);
$file = $slice->getFileUrl(1);
print_r($file);
?>
```

Ausgabe  
/banner.jpg

## ***getHtml()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$string = OOArticleSlice::getHtml();
```

### **Parameter**

keine

### **Rückgaben**

Html-Code

### **Beschreibung**

Methode; gibt den Html-Code des selektierten Slices aus.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getArticleSliceById(52);
$html = $slice->getHtml();
print_r(htmlspecialchars($html));
?>
```

Ausgabe

```
<h2>Erstellung und Pflege</h2>
```

## ***getPhp()***

### **Verfügbarkeit**

ab 2.7

### **Verwendung**

```
$string = OOArticleSlice::getPhp();
```

### **Parameter**

keine

### **Rückgaben**

PHP-Code

### **Beschreibung**

Methode; gibt den PHP-Code des selektierten Slices aus.

### **Beispiel**

```
<?
$slice = OOArticleSlice::getArticleSliceById(65);
$phpcode = $slice->getPhp();
print_r(htmlspecialchars($phpcode));
?>
```

Ausgabe

```
<? echo "<h2>getArticleSliceById()</h2>"; $slice = OOArticleSlice::getArticleSliceById(31);
print_r($slice); // Ausgabe eines Einzelwertes print <<<EOT <br><h2> Einzelwert: Value (1) </h2>
EOT; echo $slice -> getValue(1); ?>
```

## C6.04 OOMedia

Die Beispiele wurden mit der Demo der Version 3.1 erstellt.

### Übersicht:

#### Klassen Funktionen

```
$file = OOMedia::getMediaById(--MFid--)  
$file = OOMedia::getMediaByName(--MFname--)  
$file = OOMedia::getMediaByFileName(--MFname--)  
$file = OOMedia::searchMediaByFileName(--MFname--)  
$files = OOMedia::getMediaByExtension(--Ext--)*  
$files = OOMedia::searchMediaByExtension(--Ext--)*
```

#### Objekt Funktionen

```
$fileCat = $file->getCategory()  
$fileCatId = $file->getCategoryId()  
$fileCatName = $file->getCategoryName()  
$fileId = $file->getId()  
$fileParentId = $file->getParentId()  
$fileName = $file->getFileName()  
$mOrgFileName = $file->getOrgFileName()  
$fileTitle = $file->getTitle()  
$fileDescription = $file->getDescription()  
$fileCopyright = $file->getCopyright()  
$filePath = $file->getPath()  
$fileFullPath = $file->getFullPath()  
$fileWidth = $file->getWidth()  
$fileHeight = $file->getHeight()  
$fileType = $file->getType()  
$fileSize = $file->getSize()  
$fileFormattedSize = $file->getFormattedSize()  
$fileCreateUser = $file->getCreateUser()  
$fileUpdateUser = $file->getUpdateUser()  
$fileCreateDate = $file->getCreateDate(--FDate--)  
$fileUpdateDate = $file->getUpdateDate(--FDate--)  
$fileExtension = $file->getExtension()  
$fileGetIcon = $file->getIcon()  
$fileToIcon = $file->toIcon(--HtmlAtt--, --IPath--)  
$fileImage = $file->toImage(--HtmlAtt--)  
$fileHtml = $file->toHtml(--HtmlAtt--)
```

```
$fileLink = $file->toLink(--HtmlAtt--)
```

```
$fileString = $file->toString()
```

Im Folgenden werden nur die Klassenfunktionen beschrieben. Die Verwendung der Objekt-Funktionen ist vom Prinzip her in den vorhergehenden Klassen beschrieben worden.

Die Funktionen

```
tolcon(HtmlAtt, IPath)
```

```
tolmage(HtmlAtt)
```

```
toHtml(HtmlAtt)
```

```
toLink(HtmlAtt)
```

```
toString()
```

werden hier kurz erläutert, da sie aus den vorgehenden Beispielen nicht abgeleitet werden können:

[Spezielle Objekt-Funktionen](#)

## ***OOMedia-Klasse***

Verfügbarkeit

ab 3.0

Beschreibung

Klasse; Mit der OOMedia-Klasse können Sie Informationen zu den Dateien im Medienpool erhalten. Diese Daten werden in der Tabelle rex\_file gespeichert.

Dazu gehören u. a. Dateiname, -typ, -größe, Kategoriezuordnung, bei Bildern Breite und Höhe, Beschreibung, Copyright.

### ***getMediaById()***

**Verfügbarkeit**

ab 3.0

**Verwendung**

```
$media = OOMedia::getMediaById(-id-)
```

**Parameter**

File-ID

**Rückgaben**

OOMedia-Objekt ( [\_id], [\_parent\_id], [\_cat\_id], [\_cat\_name], [\_cat], [\_name],[\_orgname], [\_type], [\_size], [\_width], [\_height], [\_title], [\_description], [\_copyright], [\_updatedate], [\_createdate], [\_updateuser], [\_createuser], [\_resizeextensions])

**Beschreibung**

Methode; gibt Informationen über eine Datei mit einer bestimmten ID zurück. Jede Datei hat eine eindeutige ID.

## Beispiel

```
<?
$media = OOMedia::getMediaById(12);
print_r($media);
?>
```

## Ausgabe

```
oomedia Object(
  [_id] => 12
    [_parent_id] => 0
    [_cat_id] => 2
    [_cat_name] => Fotos
    [_cat] =>
      [_name] => pirna_aussicht.jpg
      [_orgname] => C:DOKUME~1ThomasLOKALE~1Tempphp27.tmp
      [_type] => image/jpeg
      [_size] => 10050
      [_width] => 0
      [_height] => 0
      [_title] => Pirna Aussicht
      [_description] =>
      [_copyright] => blumbeet.com
      [_updatedate] => 1133314574
      [_createdate] => 1133279891
      [_updateuser] => dergel
      [_createuser] => thomas
      [_resizeextensions] => Array
        (
          [0] => jpeg
          [1] => jpg
          [2] => gif
          [3] => png
        )
    )
)
```

Auf die einzelnen Attribute kann wieder mit den oben genannten Objekt-Funktionen zugegriffen werden.

## **getMediaByName()**

### **Verfügbarkeit**

ab 3.0

### **Verwendung**

```
$media = OOMedia::getMediaByName('name')
```

### **Parameter**

Name der Datei

### **Rückgaben**

OOMedia-Objekt ( [\_id], [\_parent\_id], [\_cat\_id], [\_cat\_name], [\_cat], [\_name],[\_orgname], [\_type], [\_size], [\_width], [\_height], [\_title], [\_description], [\_copyright], [\_updatedate], [\_createdate], [\_updateuser], [\_createuser], [\_resizeextensions])

### **Beschreibung**

Methode; gibt Informationen über eine Datei mit einem bestimmten Namen zurück.

### **Beispiel**

```
<?
```

```
$media = OOMedia::getMediaByName('briefkasten.jpg');
```

```
print_r($media);
```

```
?>
```

### **Ausgabe**

```
oomedia Object (
  [_id] => 6
    [_parent_id] => 0
    [_cat_id] => 2
    [_cat_name] => Fotos
    [_cat] =>
    [_name] => briefkasten.jpg
    [_orgname] => C:DOKUME~1ThomasLOKALE~1Tempphp10.tmp
    [_type] => image/jpeg
    [_size] => 13307
    [_width] => 0
    [_height] => 0
    [_title] => Briefkasten
    [_description] =>
    [_copyright] =>
    [_updatedate] => 1133314549
    [_createdate] => 1133276496
    [_updateuser] => dergel
    [_createuser] => thomas
    [_resizeextensions] => Array
      (
        [0] => jpeg
        [1] => jpg
        [2] => gif
        [3] => png
      )
  )
)
```

## **getMediaByFileName()**

### **Verfügbarkeit**

ab 3.1

### **Verwendung**

```
$media = $media = OOMedia::getMediaByFileName('name')
```

### **Parameter**

Name der Datei

### **Rückgaben**

OOMedia-Objekt ( [\_id], [\_parent\_id], [\_cat\_id], [\_cat\_name], [\_cat], [\_name],[\_orgname], [\_type], [\_size], [\_width], [\_height], [\_title], [\_description], [\_copyright], [\_updatedate], [\_createdate], [\_updateuser], [\_createuser], [\_resizeextensions])

### **Beschreibung**

Methode; gibt Informationen über eine Datei mit einem bestimmten Namen zurück.

### **Beispiel**

```
<?
$media = OOMedia::getMediaByFileName('briefkasten.jpg');
print_r($media);
?>
```

### **Ausgabe**

```
oomedia Object
(
    [_id] => 6
    [_parent_id] => 0
    [_cat_id] => 2
    [_cat_name] => Fotos
    [_cat] =>
    [_name] => briefkasten.jpg
    [_orgname] => C:DOKUME~1ThomasLOKALE~1Tempphp10.tmp
    [_type] => image/jpeg
    [_size] => 13307
    [_width] => 0
    [_height] => 0
    [_title] => Briefkasten
    [_description] =>
    [_copyright] =>
    [_updatedate] => 1133314549
    [_createdate] => 1133276496
    [_updateuser] => dergel
    [_createuser] => thomas
    [_resizeextensions] => Array
        (
            [0] => jpeg
            [1] => jpg
            [2] => gif
            [3] => png
        )
)
```

## **searchMediaByFileName()**

### **Verfügbarkeit**

nur 3.0 und 3.1, ab Version 3.2 gelöscht, da redundant

### **Verwendung**

```
$media = $media = OOMedia::searchMediaByFileName('name')
```

### **Parameter**

Name der Datei

### **Rückgaben**

OOMedia-Objekt ( [\_id], [\_parent\_id], [\_cat\_id], [\_cat\_name], [\_cat], [\_name],[\_orgname], [\_type], [\_size], [\_width], [\_height], [\_title], [\_description], [\_copyright], [\_updatedate], [\_createdate], [\_updateuser], [\_createuser], [\_resizeextensions])

### **Beschreibung**

Methode; gibt Informationen über eine Datei mit einem bestimmten Namen zurück.

### **Beispiel**

```
<?
$media = OOMedia::searchMediaByFileName('briefkasten.jpg');
print_r($media);
?>
```

### **Ausgabe**

```
oomedia Object
(
    [_id] => 6
    [_parent_id] => 0
    [_cat_id] => 2
    [_cat_name] => Fotos
    [_cat] =>
    [_name] => briefkasten.jpg
    [_orgname] => C:DOKUME~1ThomasLOKALE~1Tempphp10.tmp
    [_type] => image/jpeg
    [_size] => 13307
    [_width] => 0
    [_height] => 0
    [_title] => Briefkasten
    [_description] =>
    [_copyright] =>
    [_updatedate] => 1133314549
    [_createdate] => 1133276496
    [_updateuser] => dergel
    [_createuser] => thomas
    [_resizeextensions] => Array
        (
            [0] => jpeg
            [1] => jpg
            [2] => gif
            [3] => png
        )
)
```

## **getMediaByExtension()**

### **Verfügbarkeit**

ab 3.1

### **Verwendung**

```
$media = $media = OOMedia::getMediaByExtension('ext')
```

### **Parameter**

Dateiendung

### **Rückgaben**

Array (

[0] => oomedia Object (

```
[_id], [_parent_id], [_cat_id], [_cat_name], [_cat], [_name], [_orgname], [_type], [_size], [_width],
[_height], [_title], [_description], [_copyright], [_updatedate], [_createdate], [_updateuser], [_createuser],
[_resizeextensions]) )
```

[1] => oomedia Object ( ...).....

### **Beschreibung**

Methode; gibt Informationen über alle Dateien mit einer bestimmten Dateiendung zurück.

### **Beispiel**

<?

```
$media = OOMedia::getMediaByExtension('css');
print_r($media);
?>
```

### **Ausgabe**

**Array**

```
(
  [0] => oomedia Object
    (
      [_id] => 14
      [_parent_id] => 0
      [_cat_id] => 3
      [_cat_name] => PDFs
      [_cat] =>
      [_name] => blindtext.pdf
      [_orgname] => blindtext.pdf
      [_type] => application/pdf
      [_size] => 6246
      [_width] => 0
      [_height] => 0
      [_title] => Blindtext
      [_description] =>
      [_copyright] =>
      [_updatedate] => 1133314359
      [_createdate] => 1133311952
      [_updateuser] => thomas
      [_createuser] => thomas
      [_resizeextensions] => Array
        (
          [0] => jpeg
          [1] => jpg
          [2] => gif
          [3] => png
        )
    )
)
```

## ***searchMediaByExtension()***

### **Verfügbarkeit**

nur 3.1, ab Version 3.2 gelöscht, da redundant

### **Verwendung**

```
$media = OOMedia::searchMediaByExtension('ext')
```

### **Parameter**

Dateiendung

### **Rückgaben**

Array (

[0] => oomedia Object (

```
[_id], [_parent_id], [_cat_id], [_cat_name], [_cat], [_name], [_orgname], [_type], [_size], [_width],  
[_height], [_title], [_description], [_copyright], [_updatedate], [_createdate], [_updateuser], [_createuser],  
[_resizeextensions]) )
```

[1] => oomedia Object ( ...).....

### **Beschreibung**

Methode; gibt Informationen über alle Dateien mit einer bestimmten Dateiendung zurück.

### **Beispiel**

```
<?  
$media = OOMedia::searchMediaByExtension('css');  
print_r($media);  
>
```

### **Ausgabe**

#### **Array**

```
(  
  [0] => oomedia Object  
    (  
      [_id] => 14  
      [_parent_id] => 0  
      [_cat_id] => 3  
      [_cat_name] => PDFs  
      [_cat] =>  
      [_name] => blindtext.pdf  
      [_orgname] => blindtext.pdf  
      [_type] => application/pdf  
      [_size] => 6246  
      [_width] => 0  
      [_height] => 0  
      [_title] => Blindtext  
      [_description] =>  
      [_copyright] =>  
      [_updatedate] => 1133314359  
      [_createdate] => 1133311952  
      [_updateuser] => thomas  
      [_createuser] => thomas  
      [_resizeextensions] => Array  
        (  
          [0] => jpeg  
          [1] => jpg  
          [2] => gif  
          [3] => png  
        )  
    )  
)
```

## C6.05 OOMediaCategory

Die Beispiele wurden mit der Demo der Version 3.1 erstellt.

### Übersicht:

#### Klassen Funktionen

```
OOMediaCategory::getRootCategories(--On/Off--)*
$mediaCat = OOMediaCategory::getCategoryById(--MCid--)
$mediaCat = OOMediaCategory::getCategoryByName(--MCname--)
$mediaCat = OOMediaCategory::searchCategoryByName(--MCname--)
```

#### Objekt Funktionen

```
$mCatChildren = $mediaCat->getChildren()* if ($mediaCat->isRootCategory())
$mCatFiles = $mediaCat->getFiles()* if ($mediaCat->isParent())
$mCatName = $mediaCat->getName() if ($mediaCat->isValid(--Obj--))
$mCatId = $mediaCat->getId() if ($mediaCat->isHidden())
$mCatPath = $mediaCat->getPath() if ($mediaCat->hasParent())
$mCatParent = $mediaCat->getParent() if ($mediaCat->hasChildren())
$mCatParentId = $mediaCat->getParentId() if ($mediaCat->hasFilesHidden())
$mCatCreateDate = $mediaCat->getCreateDate()
$mCatUpdateDate = $mediaCat->getUpdateDate()
$mCatCreateUser = $mediaCat->getCreateUser()
$mCatUpdateUser = $mediaCat->getUpdateUser()
$mCatCountChildren = $mediaCat->countChildren()
$mCatCountFiles = $mediaCat->countFiles()
$mCatString = $mediaCat->toString()
```

Im Folgenden werden nur die Klassenfunktionen beschrieben. Die Verwendung der Objekt-Funktionen ist vom Prinzip her in den vorhergehenden Klassen beschrieben worden.

## **OOMediaCategory-Klasse**

Verfügbarkeit  
ab 3.0

Beschreibung  
Klasse; Mit der OOMediaCategory-Klasse können Sie Informationen zu den Dateikategorien im Medienpool erhalten. Diese Daten werden in der Tabelle rex\_file\_category gespeichert.

Dazu gehören u. a. Kategorienname, Erstelldatum und -aktualisierungsdatum, Name des Bearbeiters.

### **getRootCategories()**

#### **Verfügbarkeit**

ab 3.0

#### **Verwendung**

```
$media = OOMediaCategory::getRootCategories()
```

#### **Parameter**

On/Off ???

#### **Rückgaben**

Array (

[0] => oomediacy Object ( [\_id], [\_parent\_id], [\_name], [\_path], [\_hide], [\_createdate],  
[\_updatedate], [\_createuser], [\_updateuser], [\_children], [\_files])

[1] => oomediacy Object (... )

.....

)

#### **Beschreibung**

Methode; gibt Informationen zu allen Root-Kategorien aus.

#### **Beispiel**

```
<?
```

```
$media = OOMediaCategory::getRootCategories();
```

```
print_r($media);
```

```
?>
```

## Ausgabe

```
Array
(
    [0] => oomediacategory Object
        (
            [_id] => 1
            [_parent_id] => 0
            [_name] => CSS
            [_path] => |
            [_hide] => 0
            [_createdate] => 1133266929
            [_updatedate] => 1133314323
            [_createuser] => thomas
            [_updateuser] => thomas
            [_children] =>
            [_files] =>
        )

    [1] => oomediacategory Object
        (
            [_id] => 2
            [_parent_id] => 0
            [_name] => Fotos
            [_path] => |
            [_hide] => 0
            [_createdate] => 1133314329
            [_updatedate] => 1133314329
            [_createuser] => thomas
            [_updateuser] => thomas
            [_children] =>
            [_files] =>
        )

    usw. . .
```

## getCategoryById()

### Verfügbarkeit

ab 3.0

### Verwendung

```
$media = OOMediaCategory::getCategoryById(id)
```

### Parameter

Kategorie-ID

### Rückgaben

oomediacategory Object ( [\_id], [\_parent\_id], [\_name], [\_path], [\_hide], [\_createdate], [\_updatedate], [\_createuser], [\_updateuser], [\_children], [\_files])

## Beschreibung

Methode; gibt Informationen zu der Kategorie mit einer bestimmten Kategorie-ID aus.

## Beispiel

```
<?
$media = OOMediaCategory::getCategoryById(2);
print_r($media);
?>
```

## Ausgabe

```
oomediacy Object
(
    [_id] => 2
    [_parent_id] => 0
    [_name] => Fotos
    [_path] => |
    [_hide] => 0
    [_createdate] => 1133314329
    [_updatedate] => 1133314329
    [_createuser] => thomas
    [_updateuser] => thomas
    [_children] =>
    [_files] =>
)
```

## getCategoryByName()

### Verfügbarkeit

ab 3.1

### Verwendung

```
$media = OOMediaCategory::getCategoryByName('name')
```

### Parameter

Kategorienname

### Rückgaben

Array ( [0] =>

oomediacy Object ( [\_id], [\_parent\_id], [\_name], [\_path], [\_hide], [\_createdate], [\_updatedate],  
[\_createuser], [\_updateuser], [\_children], [\_files]) )

## Beschreibung

Methode; gibt Informationen zu einer Kategorie mit einem bestimmten Namen aus.

## Beispiel

```
<?
$media = OOMediaCategory::getCategoryByName('Fotos');
print_r($media);
?>
```

## Ausgabe

```
Array
(
    [0] => oomediacy Object
        (
            [_id] => 2
            [_parent_id] => 0
            [_name] => Fotos
            [_path] => |
            [_hide] => 0
            [_createdate] => 1133314329
            [_updatedate] => 1133314329
            [_createuser] => thomas
            [_updateuser] => thomas
            [_children] =>
            [_files] =>
        )
)
```

## searchCategoryByName()

### Verfügbarkeit

ab 3.0

### Verwendung

```
$media = OOMediaCategory::searchCategoryByName('name')
```

### Parameter

Kategorienname

### Rückgaben

Array ( [0] =>

oomediacy Object ( [\_id], [\_parent\_id], [\_name], [\_path], [\_hide], [\_createdate], [\_updatedate],  
[\_createuser], [\_updateuser], [\_children], [\_files]) )

### Beschreibung

Methode; gibt Informationen zu einer Kategorie mit einem bestimmten Namen aus.

## Beispiel

```
<?
$media = OOMediaCategory::searchCategoryByName('Fotos');
print_r($media);
?>
```

## Ausgabe

```
Array
(
    [0] => oomediacategory Object
        (
            [_id] => 2
            [_parent_id] => 0
            [_name] => Fotos
            [_path] => |
            [_hide] => 0
            [_createdate] => 1133314329
            [_updatedate] => 1133314329
            [_createuser] => thomas
            [_updateuser] => thomas
            [_children] =>
            [_files] =>
        )
)
```

## C7. Import & Export

Ein Import besteht in der Regel aus dem Import der Datenbank (Tabellen und Inhalte) und den Dateien, die beim Export erstellt wurden. Dabei werden in der Regel die Dateien aus den Ordnern files, css, js und pics gesichert. (Mehr unter Export)

### Import

Auf der linken Seite können Sie unter **Datenbankimport** die Tabellen und deren Inhalte importieren. Dazu können Sie entweder die Redaxo-Import Dateien auswählen und importieren, oder die darunter angegebenen Importe importieren. Dabei werden die bisherigen Inhalte der Datenbank gelöscht und überschrieben!

Danach können Sie unter **Dateienimport** die benötigten Dateien importieren. Nach erfolgreichem Import werden alle Artikel sowie der Cache neu generiert. Wechseln Sie danach in die Strukturverwaltung oder auf die Webseite und sehen das Ergebnis.

### Export

Auf der rechten Seite finden Sie Einstellungen und Optionen für den Export der Tabellen und Dateien ihrer Webseite.

Für Sicherung der Datenbank, wählen Sie erste Option und entscheiden unten, ob Sie die Export-Datei auf dem Server speichern oder auf Ihrer Festplatte sichern wollen.

Wählen Sie die Option **Auf dem Server speichern** wird die Datei auf dem Server gespeichert und auf der linken Seite, unter **Import** angezeigt. Sie finden die auf dem Server gespeicherte Datei im Ordner "redaxoincludeaddonsimport\_exportfiles".

## C8. CVS

Via CVS können sich Entwickler die aktuellsten Versionen von REDAXO herunterladen. Diese Version kann allerdings Fehler enthalten! Sie wird **nicht** für den Produktiv-Einsatz empfohlen!

Zugangsdaten

### Redaxo 2.7

```
cvs -z3 -d:pserver:anonymous@cvs.redaxo.berlios.de:/cvsroot/redaxo export redaxo2_7
```

### Redaxo 3.0

```
cvs -z3 -d:pserver:anonymous@cvs.redaxo.berlios.de:/cvsroot/redaxo export redaxo3_0
```

Berlios CVS Online Browser

### REDAXO3.x

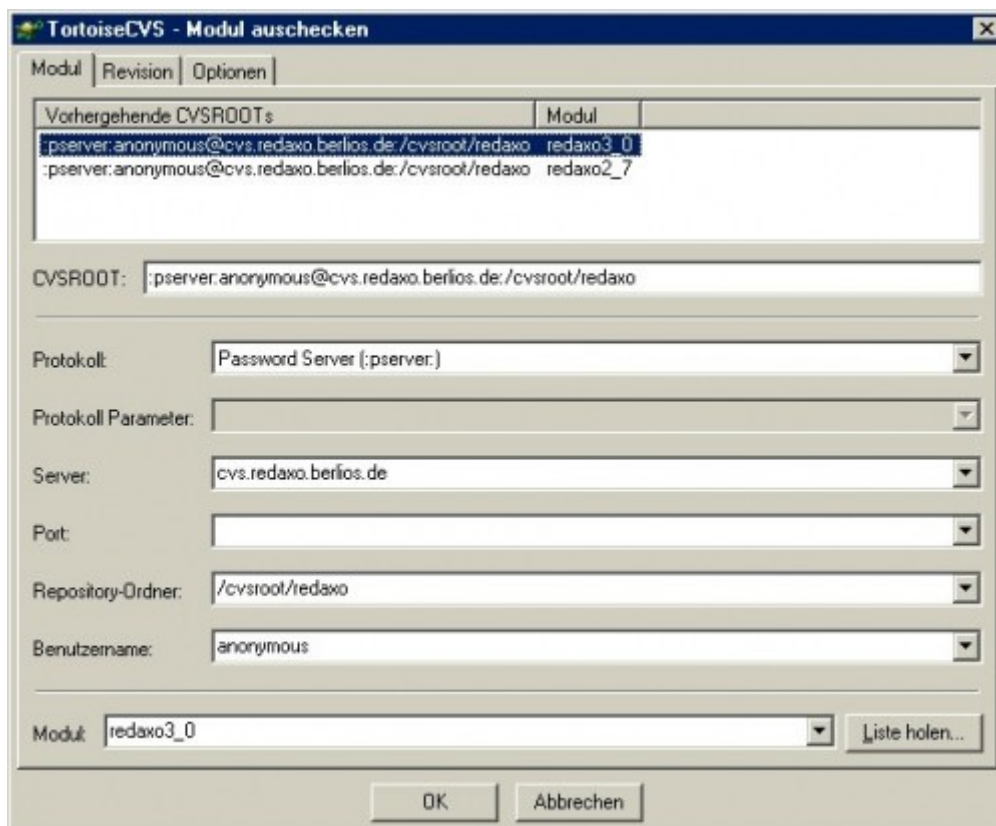
[http://cvs.berlios.de/cgi-bin/viewcvs.cgi/redaxo/redaxo3\\_0/](http://cvs.berlios.de/cgi-bin/viewcvs.cgi/redaxo/redaxo3_0/)

### REDAXO2.x

[http://cvs.berlios.de/cgi-bin/viewcvs.cgi/redaxo/redaxo2\\_7/](http://cvs.berlios.de/cgi-bin/viewcvs.cgi/redaxo/redaxo2_7/)

### Tortoise CVS Client (Windows)

Die Einstellungen für TortoiseCVS können sie dem folgenden Screenshot entnehmen:



### Hinweis:

Eine evtl. vorhande Firewall oder ein Router muss entsprechend konfiguriert werden!

## **CVS unter Linux**

Informationen, wie sie über [Linux das CVS](#) nutzen können finden Sie auf [Berlios.de](#)

### **Ergänzende Informationen finden sich unter...**

<http://www.xwolf.de/artikel/cvstutor>

<http://forum.redaxo.de/viewtopic.php?t=940>

<http://www.tortoise cvs.org/>

## C.9 Extension Points

REDAXO bietet durch eine Extension Point API die Möglichkeit, via Addon/Modul/Template direkt in den Workflow einzugreifen. Dies ist mit folgenden Funktionen möglich:

- [rex\\_register\\_extension\\_point](#)
- [rex\\_register\\_extension](#)
- [rex\\_is\\_registered\\_extension](#)
- 

Diese Extension Points kann man sich als Schnittstellen vorstellen, an denen man sich mit einer beliebigen Funktion einhaken kann. Jeder Extension Point hat unterschiedliche Parameter, die der eingehakten Funktion als Array übergeben werden.

Bereits in REDAXO implementiert sind folgende Extension Points:

- [OUTPUT\\_FILTER](#)
- [OUTPUT\\_FILTER\\_CACHE](#)
- [ADDONS\\_INCLUDED](#)
- [ALL\\_GENERATED](#)
- [URL\\_REWRITE](#)
- [MEDIA\\_ADDED, MEDIA\\_UPDATED](#)
- [CAT\\_ADDED, CAT\\_UPDATED, CAT\\_DELETED, CAT\\_STATUS](#)
- [ART\\_ADDED, ART\\_UPDATED, ART\\_DELETED, ART\\_STATUS](#)
- [CLANG\\_ADDED, CLANG\\_UPDATED, CLANG\\_DELETED](#)

### C9.02 Beispielcode

Mit folgendem Code, kann eine Manipulation des Seiteninhaltes erreicht werden. Dazu wird der Extension Point [OUTPUT\\_FILTER](#) verwendet:

```
<?php
// Die Funktion "rex_insert_special_urls" an dem
// Extension Point "OUTPUT_FILTER" einhaken.
rex_register_extension('OUTPUT_FILTER', 'rex_insert_special_urls');

// Funktion, zur Ersetzung von Speziellen Namen
// durch verweise auf deren Homepage
function rex_insert_special_urls($params)
{
$content = $params['subject'];
$search = array(
'REDAXO',
'phpBB'
);
$replace = array(
'<a href="http://www.redaxo.de">REDAXO</a>',
'<a href="http://www.phpbb.de">phpBB</a>'
);
return str_replace( $search, $replace, $content);
}
?>
```